πby3

π - Ingest

Oracle Source CDC Connector – Snowflake Native App

Attribute	Details
Version	01
Author	Pibythree Developers
Product	Pi-Ingest (Oracle CDC)
Platform	Snowflake Native App with Snowpark Container Services
Document Type	Technical & Functional Specification

Table of Contents

1. Introduction 5
1.1 Purpose 5
1.2 Intended Audience 5
1.3 Document Conventions 6
1.4 Solution Summary 6
2 Key Considerations 8
2.1 Scope 8
2.2 Out of Scope 8
2.3 Assumptions 8
2.4 Dependencies 8
2.5 Risks 9
2.6 Non-Functional Requirements 9
2.7 Constraints 10
3 Business Feature Details 11
3.1 Business Process Flow 11
3.2 RBAC & UBAC 11
3.3 CDC Processing 11
3.4 Monitoring & Operational Visibility 11
3.5 Business Use Cases 11
3.6 Business Rules 12
3.7 Success Metrics and KPIs 12
4 Technical Feature Details 13
4.1 Technical Architecture 13
4.2 RBAC & UBAC 13
4.3 Logging & Monitoring 13
4.4 Secrets & Secure Access Management 13
4.4.1 Oracle Secrets Management 14
4.4.2 Network Rule Configuration 14
4.4.3 External Access Integration 14
4.4.4 Secure Function for Credential Access 14
4.5 End-to-End Data Flow 15
4.6 Job Orchestration 15
4.7 Load Design 16
4.8 Dynamic Chunk Size Calculation 16
4.9 Error Handling and Retry 16
4.10 Performance and Scalability Considerations 16
4.11 Security and Compliance 17
5 Technical Details - Backend 18
5.1 Database Details 18
5.1.1 Database Details in Snowflake Environment 18
5.2 Schema Details 18
5.2.1 Schema Details in Snowflake Environment 18
5.3 Table Details 19
5.3.1 Table Details in Snowflake Environment 19
5.4 Objects Details 21
5.4.1 Object Details in Snowflake Environment 21
5.5 Snowflake Container Services (SPCS) Deployment 22

5.5.1 Compute Pool Configuration 23
5.5.2 Snowflake Service Details 23
5.6 Oracle to Snowflake Data Type Mapping 24
5.7 Extraction Logic 25
5.8 Representative Table Structures 26
5.8.1 CONFIG_ORC_INGEST.CDC_JOB_REGISTRY ..... **Error! Bookmark not defined.**
5.8.2 CONFIG_ORC_INGEST.CDC_CONFIG ..... **Error! Bookmark not defined.**
5.8.3 LOGS_ORC_INGEST.CDC_CHECKPOINT ..... **Error! Bookmark not defined.**
5.8.4 LOGS_PG_INGEST.ERROR_LOG ..... **Error! Bookmark not defined.**
5.9 Application Roles and Privileges 30
6 Oracle Source Preparation 32
6.1 Supported Environments 32
6.2 Create a Dedicated CDC User 32
6.3 Preparing Source Database 34
6.4 Network and Authentication 35
6.4 Recommended Server Settings 35
6.5 Source Readiness Checklist 36
7 Snowflake Setup and Configuration 37
7.1 Create the Network Rule 37
7.2 Create the Secret 37
7.3 Create the External Access Integration 37
7.4 Application Privileges and Reference Binding 37
7.5 Configure the Source 37
7.6 Register Tables for Replication 38
8 Operations and Runbook 39
8.1 Daily Operational Checks 39
8.2 Routine Operations 40
8.3 Useful Diagnostic Queries 40
8.4 Cost Management 41
8.5 Backup and Recovery 41
9 Testing and Validation 43
9.1 Test Approach 43
9.2 Test Scenarios 43
9.3 Acceptance Criteria 44
9.4 Go-Live Checklist 44
10 Troubleshooting Scenarios & Recommendations 46
10.1 Additional Error Reference 47
10.2 Escalation Path 48
11 Glossary 49
12 Reference Artifacts & Editable Links 50
13 Appendix 52
13.1 Usage Guidelines for Contributors 52
13.2 Environment Deployment Guide 52
13.2.2 Deployment Steps 52
13.2.3 Environment Configuration 53
13.2.4 Validation 53
13.3 Frequently Asked Questions 54
13.4 Roadmap Considerations 55
13.5 Revision History 55

13.6 Document Review and Sign-off 55
14 Support & Contact Information 56
14.1 Technical Support 56
14.2 Business & Sales Inquiries 56
14.3 Documentation & Feedback 56
14.4 Customization & Full Product 56

1. Introduction

This document outlines the requirements, architecture, and implementation details for the π-Ingest: Oracle Change Data Capture (CDC) Source Connector. The objective of this initiative is to deliver a secure, scalable, and reliable log-based data ingestion framework—distributed as a Snowflake Native App—that captures data changes from Oracle databases and replicates them into the consumer's Snowflake account for downstream analytics, operational monitoring, and data synchronization.

The document serves as a single source of truth for all stakeholders involved in the project, including business users, data engineers, solution architects, Oracle Database Administrators, testers, and project managers. It captures the functional and non-functional requirements, business workflows, technical architecture, security controls, and backend implementation approaches within the Snowflake Native App Framework.

The π-Ingest solution leverages the Snowflake Native App distribution model and high-performance capabilities such as Snowpark Container Services (SPCS), Snowflake Image Repositories, OpenLogReplicator (OLR), automated SFTP-based archive log transfer, and Reference Binding for secure credential management and external access. The solution captures DML and supported DDL changes from archived redo logs, processes them through coordinated containerized services, and applies the resulting changes to Snowflake target tables. The application logic executes within the consumer's Snowflake environment, supporting data governance, secure credential isolation, transactional consistency, and operational observability.

This initiative is part of the broader π-Snow roadmap and the πby3 Ingestion Suite. It follows the common Native App framework and configuration-driven operational model used across the Pi-Ingest connector ecosystem. Oracle-specific components include redo and archive log processing through OpenLogReplicator, automated SFTP-based archive log shipping, Oracle credential resolution, and Oracle-to-Snowflake data type mapping. This architecture provides a foundation for continuous change data capture, checkpoint-based recovery, auditability, and automated schema evolution, while minimizing manual configuration and operational overhead for the end user.

1.1 Purpose

The purpose of this document is to describe, in sufficient detail, how **π-Ingest ingests change data from Oracle databases into Snowflake** using a log-based Change Data Capture (CDC) architecture. It defines the business objectives, technical design, Snowflake objects created by the application, steps required to prepare the Oracle source database and the Snowflake consumer account, operational procedures for deploying and running the solution, and validation approaches used to confirm that the deployment is healthy.

The document also describes the roles of OpenLogReplicator (OLR), the SFTP-based archive log transfer component, and the Snowflake containerized CDC consumer in capturing, processing, and applying Oracle database changes. It covers configuration and reference binding, security and connectivity requirements, checkpoint management, audit logging, error handling, supported schema evolution, and recovery procedures to support reliable and continuous data replication.

1.2 Intended Audience

Audience	How this document is used

Audience	How this document is used
Business Users / Data Owners	Understand what Oracle data is ingested, how database changes are replicated into Snowflake, and how data quality, replication freshness, and ingestion status can be monitored.
Data Engineers	Configure CDC jobs and table filters, operate the ingestion services, monitor replication progress, and troubleshoot ingestion failures.
Solution Architects	Review the end-to-end CDC architecture, scalability, security posture, recovery mechanisms, and integration with the wider Snowflake environment.
Snowflake / Security Administrators	Review application privileges, Native App reference bindings, secrets management, network rules, External Access Integration (EAI), and Snowpark Container Services (SPCS) configuration.
Oracle DBAs	Prepare the Oracle source database, configure required logging and supplemental logging, provision source credentials and privileges, establish network and SFTP access, and monitor the impact of CDC on source database resources.
Testers / QA	Execute deployment validation checklists and CDC test scenarios, verify INSERT, UPDATE, and DELETE replication, validate supported DDL handling, and certify the deployment against defined acceptance criteria.

1.3 Document Conventions

- **Placeholders:** Values shown in angle brackets, for example `<oracle_host>`, must be replaced with environment-specific values.
- **Object names:** Snowflake and Oracle object names are shown in uppercase or monospaced style when referring to specific objects.
- **Provider / Consumer:** The Provider builds and publishes the application; the Consumer installs and operates it in their own Snowflake account.
- **Representative structures:** Table layouts, SQL statements, and configurations illustrate the design and must be aligned with the deployed application version before use.

1.4 Solution Summary

Characteristic	Description
Source	Oracle Database 11g, 12c, 19c, 21c, and 23c (EE / SE / XE, subject to compatibility).
Target	Snowflake with Snowpark Container Services (SPCS) enabled.
Distribution model	Snowflake Native App delivered via Marketplace or Private Listing.
Execution engine	OpenLogReplicator, CDC consumer, and SFTP-based archive log streamer running in SPCS containers.
Load types	Log-based continuous CDC for capturing and replicating Oracle database changes.

Characteristic	Description
Processing	Redo and archived redo log processing, with checkpoint-based recovery.
Security	Consumer-owned secrets, Reference Binding, Network Rules, External Access Integration, and application roles.
Observability	CDC audit logs, error logs, checkpoints, replication progress, and container logs.

2 Key Considerations

This section outlines the foundational elements that influence the scope, planning, and execution of the π-Ingest Oracle CDC project, including scope boundaries, assumptions, dependencies, and risks.

2.1 Scope

The scope of this project includes the following:

- Real-time change data capture (CDC) from Oracle databases into Snowflake.
- Incremental replication of inserts, updates and deletes (soft deletes) into Snowflake tables.
- Automatic handling of Oracle schema changes.
- Execution of ingestion and apply jobs using Snowpark Container Services (SPCS).
- Centralized logging and monitoring of ingestion and apply job execution.

2.2 Out of Scope

- Data ingestion from sources other than Oracle.
- Advanced analytics, reporting, or business dashboarding.
- Unsupported or destructive schema changes beyond the documented DDL evolution capabilities.

2.3 Assumptions

Sr. No	Description	Implications, if not met
1	Active Oracle database with required privileges, logging configuration, and network access.	CDC cannot capture the required database changes.
2	Snowflake account with SPCS enabled.	Application cannot be deployed or executed.
3	Required Snowflake roles and permissions are available.	Application objects and services cannot be created or operated.
4	Snowflake network access to the Oracle listener and SFTP endpoint is configured through the required network rules and External Access Integration.	External connectivity will fail.
5	Oracle connectivity, credentials, archive log availability, and required supplemental logging are configured.	Connection failures or incomplete CDC capture may occur.

2.4 Dependencies

Sr. No	Description	Owner	Implications, if not met
1	Oracle credentials and source connection details (user, password, host, port, service name).	Oracle DBA / Admin Team	CDC connectivity cannot be established.
2	Application installation and required account-level privileges.	Consumer AccountAdmin	Application infrastructure cannot be provisioned or operated.
3	Security approval and Reference Binding for secrets and External Access Integration.	Consumer AccountAdmin	Application cannot connect to Oracle or the SFTP endpoint.

Sr. No	Description	Owner	Implications, if not met
4	Compute pool and warehouse availability, where required by the application configuration.	Consumer AccountAdmin	Service execution delays or failures.

2.5 Risks

Sr. No	Description	Mitigation Strategy
1	Credential exposure or misconfiguration.	Use Snowflake Secrets and Reference Binding to protect credentials within the consumer account.
2	Consumer region incompatibility for SPCS.	Validate SPCS availability in the target consumer region before installation.
3	Load impact on the Oracle production database.	Configure appropriate log retention, monitor source impact, and ensure required redo and archive logs remain available.
4	Data type incompatibility between Oracle and Snowflake.	Apply documented Oracle-to-Snowflake type mappings and handle unsupported types explicitly.

2.6 Non-Functional Requirements

Category	Requirement	Approach
Security	Credentials must remain confidential and never be hardcoded.	Consumer-owned Snowflake Secrets and Reference Binding.
Performance	Process CDC changes efficiently without exhausting container resources.	Bounded queues, configurable batching, and transaction-aligned processing.
Scalability	Support multiple tables and continuous change processing.	Metadata-driven configuration and containerized services.
Reliability	Recover from failures without losing or duplicating committed changes.	Checkpoints, idempotent processing, and audit logging.
Observability	Every job and change-processing batch must be traceable.	CDC audit tables, checkpoint records, error logs, and container logs.
Maintainability	Adding supported tables should require minimal code changes.	Configuration-driven table selection and filtering.
Governance	Consumers must control application access and connectivity.	Application roles, privileges, and External Access Integration controls.
Portability	Support future relational database connectors.	Separate source-specific capture components from shared Snowflake ingestion and logging components.

2.7 Constraints

- **Source Load:** CDC processing reads Oracle redo and archived redo logs; configure appropriate log retention and monitor source database impact.
- **Network:** Snowflake must reach the Oracle listener and SFTP endpoint through approved network rules and External Access Integration.
- **Snowflake Edition:** SPCS and External Access Integration availability depends on the consumer account's supported edition and region.
- **Schema Evolution:** Oracle DDL changes must be evaluated against supported schema evolution and Oracle-to-Snowflake data type mappings.
- **Log Availability:** Required redo and archived redo logs must remain accessible and available for the SCN range being processed; missing logs can interrupt CDC recovery.

3 Business Feature Details

This section outlines the key business functionalities delivered by the π-Ingest Oracle CDC solution, ensuring reliable, secure, and timely replication of Oracle database changes into Snowflake.

3.1 Business Process Flow

The solution captures Oracle database changes from redo and archived redo logs using OpenLogReplicator. An SFTP-based streamer transfers archive logs to a Snowflake internal stage, where the OLR publisher processes them and streams change events to the CDC consumer. The consumer applies changes to Snowflake target tables and maintains checkpoints and audit records.

3.2 RBAC & UBAC

Role Based Access Control (RBAC) and User Based Access Control (UBAC) are implemented to ensure secure and governed access to data within Snowflake.

RBAC

- Access is controlled through application roles and Snowflake privileges.
- Roles govern access to application objects, configuration, target data, and operational resources.

UBAC

- User access is mapped to roles according to business responsibilities.
- Privileges follow the principle of least privilege.

3.3 CDC Processing

- **Change Capture:** Captures supported Oracle DML changes from redo and archived redo logs.
- **Continuous Processing:** Processes change events and applies them to Snowflake target tables.
- **Recovery:** Checkpoints support resuming processing from the last confirmed position.
- **Failure Handling:** Audit records and logs support troubleshooting and recovery.

3.4 Monitoring & Operational Visibility

The solution includes monitoring and logging capabilities to provide visibility into ingestion and transformation activities.

- The solution provides visibility into CDC processing through:
- Checkpoint and replication progress tracking.
- Audit records for processed changes and batches.
- Error logs and container logs for troubleshooting.
- Monitoring of service health and processing failures.

3.5 Business Use Cases

Use Case	Description	Typical Schedule
Operational reporting	Make Oracle transactional data available in Snowflake for reporting without querying the production database.	Continuous CDC
Data warehouse consolidation	Combine Oracle application data with other enterprise sources in a unified analytical platform.	Continuous CDC

Use Case	Description	Typical Schedule
Near-real-time monitoring	Keep frequently changing tables close to current for operational teams.	Continuous CDC
Historical archiving	Retain captured data changes in Snowflake independently of source log retention.	Continuous CDC with target retention policies
Data science enablement	Provide query-ready Oracle datasets for feature engineering and model training.	Continuous CDC
Migration and replatforming	Capture source changes to support data migration into Snowflake.	Initial load and CDC, subject to implementation support

3.6 Business Rules

- **Source of Truth:** Oracle remains the system of record; Snowflake stores the replicated data.
- **Table Selection:** Only configured and supported tables are processed; no tables are captured implicitly.
- **Change Capture:** Supported DML changes are captured from redo and archived redo logs, subject to logging configuration and log availability.
- **Failure Visibility:** Processing failures are recorded in audit and error logs to support troubleshooting and recovery.
- **Access:** Access to configuration, target data, and operational logs is controlled through Snowflake roles and privileges.
- **Retention:** Checkpoints, audit records, and error logs support recovery and operational troubleshooting, subject to consumer retention policies.

3.7 Success Metrics and KPIs

KPI	Definition	Where Measured
CDC processing health	Whether CDC services are running and processing changes.	SPCS service status and container logs
Data freshness	Delay between source changes and their application in Snowflake.	CDC checkpoints and audit records
Processing volume	Number of change events and batches processed.	CDC_AUDIT_EVENT, CDC_AUDIT_BATCH
Processing latency	Elapsed time for change processing and target application.	Audit records and container logs
Error rate	Number and types of processing errors over time.	CDC_DEAD_LETTER and container logs
Recovery progress	Latest confirmed SCN and processing position.	CDC_CHECKPOINT

4 Technical Feature Details

This section outlines the technical implementation of the Oracle CDC solution. It translates business requirements into technical components within the Snowflake Native App Framework, covering system architecture, CDC processing, data modeling, security, and integration with Snowflake-native compute resources.

4.1 Technical Architecture

The solution follows a Snowflake Native App architecture, where all application logic, container images, and database objects are bundled into an Application Package and installed directly into the consumer's Snowflake account.

Fig. π-Ingest (Oracle CDC) Architecture

- **Oracle Connectivity:** Connects to the Oracle source database and processes redo and archived redo logs.
- **Execution Engine:** OpenLogReplicator, the CDC consumer, and the archive log streamer run as containerized services in Snowpark Container Services (SPCS).
- **Secure Credential Management:** Oracle and SFTP credentials are managed through Snowflake Secrets and Reference Binding.
- **Log Processing:** OpenLogReplicator parses Oracle redo and archived redo logs and streams change events to the CDC consumer.
- **Data Processing:** The CDC consumer applies captured changes to configured Snowflake target tables and maintains checkpoints and audit records.
- **Automated Provisioning:** Application procedures and configuration tables support service deployment and runtime configuration.

4.2 RBAC & UBAC

Access control is implemented using Application Roles defined within the Snowflake Native App and mapped to consumer account roles to ensure governed access.

- **Application Roles:** Control access to application schemas, stored procedures, and resources.
- **Consumer Mapping:** The consumer administrator grants Application Roles to appropriate account-level roles.
- **Data Isolation:** Configuration, target data, and operational logs are protected through Snowflake privileges, ensuring access is limited to authorized users.

4.3 Logging & Monitoring

Logging and monitoring track CDC processing, service health, and operational issues.

- **Audit Logging:** CDC processing status and audit records are maintained in the `LOGS_ORC_INGEST` schema.
- **Progress Tracking:** Checkpoints and audit records track processing progress and captured changes.
- **Error Handling:** Error records and container logs support troubleshooting of connection failures, log-processing errors, and service interruptions.

4.4 Secrets & Secure Access Management

This solution implements secure credential management and controls external connectivity to Oracle and SFTP endpoints without exposing sensitive information in application code or to the application provider.

- **Consumer-Owned Secrets:** Oracle and SFTP credentials are managed through Snowflake Secrets and Reference Binding within the consumer account.

- **External Access Integration (EAI):** Outbound connectivity to Oracle and SFTP is controlled through approved network rules and EAI configuration.
- **Secure Runtime Access:** Bound secrets are accessed at runtime by authorized application components.
- **Zero-Hardcoding Policy:** Credentials and sensitive authentication details are not hardcoded in the application package.
- **RBAC-Governed Security:** Access to application resources and security integrations is governed through application roles and consumer account privileges.

4.4.1 Oracle Secrets Management

- **Secret Type:** Oracle and SFTP credentials are stored using Snowflake Secrets, with `GENERIC_STRING` used for the SFTP private key where applicable.
- **Payload:** Oracle credentials include username and password; connection details such as host, port, and service name are configured separately.
- **Provider Isolation:** Secrets are bound to the application through references and accessed at runtime without exposing raw values to the application provider.
- **Access Control:** Secret creation and management are restricted by the consumer's Snowflake RBAC and granted privileges.

4.4.2 Network Rule Configuration

- **Egress Control:** Egress network rules (MODE = EGRESS, TYPE = HOST_PORT) allow outbound access only to the approved Oracle host and ports.
- **Whitelisting:**
- Two Oracle endpoints are explicitly whitelisted to prevent unauthorized data egress:
  1. the Oracle listener (your_host:1521), used by OpenLogReplicator
  2. the SSH/SFTP endpoint (your_host:22), used to copy the archive logs
- **Granular Security:** Through the external access integration (oracle_cdc_eai), the application's container services can communicate only with these verified Oracle endpoints. They can use only the two allowed secrets: oracle_credentials and sftp_pem_key.

4.4.3 External Access Integration

- **Reference Binding:** The application requests access to consumer-defined External Access Integrations (EAI) through Reference Binding, allowing the consumer to approve and govern external connectivity.
- **Security Association:** The EAI links approved network rules for Oracle and SFTP connectivity into a controlled security context.
- **Restricted Execution:** Only authorized application services configured to use the integration can access the approved external endpoints.
- **Governance:** EAI usage is governed by the consumer's Snowflake RBAC and privileges, allowing the consumer to control or revoke external connectivity.

4.4.4 Secure Function for Credential Access

- **Secure Secret Access:** Secure functions reference Snowflake Secrets to retrieve Oracle and SFTP credentials internally.
- **External Connectivity:** External Access Integration (EAI) enables outbound connections to approved Oracle and SFTP endpoints where required.

- **Credential Protection:** The function provides connection parameters to authorized services without exposing raw credentials.
- **Execution Control:** Only authorized roles with the required privileges can access or execute the secure function.

4.5 End-to-End Data Flow

The following steps describe what happens in a single scheduled run of Pi-Ingest CDC for Oracle.

Step	Stage	Description
1	Trigger	The Snowflake Native App starts or resumes the SPCS CDC service through the configured service lifecycle and stored procedures.
2	Configuration Read	The service reads CDC_JOB_REGISTRY, CDC_CONFIG, OLR_CONFIG, TABLE_FILTER_CONFIG, and FSTREAM_CONFIG to determine job settings, source configuration, and table filters.
3	Credential Retrieval	Oracle and SFTP credentials are accessed at runtime through bound Snowflake Secrets and held in memory as required.
4	Connection	The services connect to Oracle and SFTP through approved network rules and External Access Integrations (EAI).
5	Log Acquisition	The ec2-ingest service transfers Oracle archive logs to the Snowflake internal stage mounted at /snowflake_stage.
6	CDC Capture	OpenLogReplicator (olr-publisher) processes available redo/archive logs and streams change events to cdc-consumer over 127.0.0.1:5555.
7	Event Processing	The changes are split per table along transaction boundaries. Each key is reduced to its last change before the MERGE.
8	Target Merge	Changes are applied to configured target tables in ORC_INGEST, including insert, update, delete, and supported schema changes.
9	Audit and Reconciliation	Processing outcomes and batch details are recorded in CDC audit tables; errors are recorded for troubleshooting.
10	Checkpoint Update	The table's position is updated in the same transaction as the MERGE. The global position (LWM) is saved before OLR is told it can move on.

4.6 Job Orchestration

- **SPCS Service Scheduling:** CDC services run according to the configured service lifecycle and orchestration procedures; scheduled execution depends on the deployed configuration.
- **Continuous CDC Processing:** The Oracle CDC services process available redo/archive logs and stream changes to Snowflake targets.
- **Concurrency:** Queue limits and batch-size settings control event buffering and processing. Oracle log capture and target application operate through separate services.

- **Lifecycle Control:** Service start, suspend, and resume operations are managed through authorized application procedures and Snowflake privileges.
- **Dependencies:** Target changes and checkpoint updates depend on successful CDC processing. Downstream consumers can use CDC audit and checkpoint records to assess processing progress.

4.7 Load Design

- **The connector is CDC only**. There is no snapshot or initial load.
- **Start position**: CDC_CONFIG.OLR_START (now or scn:<n>) and OLR_CONFIG.START_SCN. A position already saved in CDC_APPLY_STATE takes priority over both. FSTREAM_CONFIG.START_FROM_FILE decides which archive files are copied.
- **Idempotency**: each target row stores _CDC_SEQ, and a row only changes when a newer change arrives.

4.8 Dynamic Chunk Size Calculation

- **Dynamic Batching:** CDC batch sizes are controlled through runtime configuration rather than calculated from source table row and column counts.
- **Inputs:** Batch processing uses configured limits such as `BATCH_MAX_ROWS` and `BATCH_MAX_BYTES`.
- **Principle:** Batches are constrained by row-count and byte-size limits to manage memory usage and processing throughput.
- **Bounds:** Queue capacity and batch limits are configurable to prevent unbounded buffering and excessive memory consumption.
- **Observability:** Batch processing metrics and outcomes are recorded in CDC audit tables and service logs for performance analysis.

4.9 Error Handling and Retry

Failure Scenario	Handling	Outcome
Oracle or SFTP connection failure	Connection errors are logged; retry behavior depends on the service implementation.	Processing may stop until connectivity is restored.
CDC processing error	The consumer records processing errors and continues only where recovery is safe.	Failed events may require investigation or dead-letter handling.
Data type conversion error	The error is recorded in service logs or error records where implemented.	The affected change may fail; checkpoint advancement must follow successful processing.
Container crash	SPCS service behavior follows its configured lifecycle and restart policy.	Processing can resume from the last confirmed checkpoint, provided the required logs remain available.
Reference or privilege error	The service reports access or configuration errors in its logs.	The consumer must correct the binding or privileges before processing can continue.

4.10 Performance and Scalability Considerations

- **SPCS Compute Sizing:** Configure the compute pool and service resource limits according to CDC workload, log volume, and processing requirements.

- **Oracle Log Availability:** Maintain sufficient redo/archive log retention to support continuous processing and recovery from the last confirmed checkpoint.
- **Network Connectivity:** Ensure reliable connectivity between Oracle, SFTP, and Snowflake services through approved network rules and External Access Integrations.
- **Warehouse Usage:** Configure the warehouse used by the CDC consumer for target-table operations and audit updates, where applicable.
- **Batch Processing:** Tune queue capacity and batch row/byte limits to balance throughput, memory usage, and processing latency.
- **Parallelism:** Scale service resources and concurrent processing cautiously while monitoring Oracle load, SPCS resource utilization, and CDC latency.

4.11 Security and Compliance

- **Least Privilege:** Oracle credentials use the minimum required privileges for CDC capture, while Snowflake roles restrict access to application resources and target data.
- **Encryption in Transit:** Secure Oracle and SFTP connectivity according to the configured network and endpoint security requirements.
- **Encryption at Rest:** Snowflake data is encrypted by the platform; encryption and protection of Oracle source storage remain the source administrator's responsibility.
- **Auditability:** CDC processing outcomes, checkpoints, errors, and batch activity are recorded in the configured logging tables and service logs.
- **Data Isolation:** CDC data is stored in the consumer's Snowflake account. Consumer-owned secrets and access controls help prevent provider access to credentials and data.
- **Secret Rotation:** Oracle and SFTP credentials can be updated through Snowflake Secrets and their bindings, subject to the deployed configuration.
- **Sensitive Columns:** Access to sensitive target columns can be controlled through Snowflake privileges and masking policies. Column exclusion depends on the supported table-filter configuration.

5 Technical Details - Backend

The secure, versioned instance of the π-Ingest Oracle CDC Source Connector installed in the consumer's Snowflake account encapsulates the internal schemas used to store CDC configuration, operational logs, staged Oracle archive logs, checkpoints, audit records, and replicated Oracle datasets.

5.1 Database Details

5.1.1 Database Details in Snowflake Environment

Sr. No	Object Name	Object Type	Description
1	ORC_INGEST_APP	Application Database	The secure, versioned π-Ingest Oracle CDC Source Connector installed in the consumer's Snowflake account. It encapsulates schemas for CDC configuration, operational logs, staged Oracle archive logs, checkpoints, audit records, and replicated Oracle datasets.

5.2 Schema Details

5.2.1 Schema Details in Snowflake Environment

Sr. No	Layer	Object Name	Object Type	Description
1	Application	CORE	SCHEMA	Internal application schema containing the lifecycle, reference-callback and credential procedures, and the Oracle CDC SPCS service.
2	Configuration	CONFIG_ORC_INGEST	SCHEMA	Internal application schema storing the CDC job registry, writer settings, OpenLogReplicator settings, the tables to replicate, and SFTP file-transfer settings.
3	Logs	LOGS_ORC_INGEST	SCHEMA	Internal application schema storing apply positions and fencing state, batch logs, service events, DDL logs, dead-letter records, file-transfer audit and errors, the OpenLogReplicator file log, and monitoring views.
4	Landing	LANDING_ORC_INGEST	SCHEMA	Internal staging schema containing the Snowflake stage that holds Oracle archive logs and OpenLogReplicator checkpoints.
5	Target	ORC_INGEST	SCHEMA	Target schema containing the replicated Oracle tables, one row per primary key, with pipeline metadata columns for soft deletes, change ordering and timestamps.

5.3 Table Details

5.3.1 Table Details in Snowflake Environment

Sr. No	Layer	Object Name	Object Type	Description
1	Configuration	CONFIG_ORC_INGEST.CDC_JOB_REGISTRY	TABLE	Registry of CDC jobs: job ID, name, description and active flag.
2	Configuration	CONFIG_ORC_INGEST.CDC_CONFIG	TABLE	Table-writer settings for each job: warehouse, database and schemas, OpenLogReplicator host and port, start position (OLR_START), queue and batch limits (MAX_QUEUE_COUNT, MAX_QUEUE_BYTES, APPLY_MAX_EVENTS, TABLE_INBOX_MAX), APPLY_MODE, confirm and reconnect intervals, and alert threshold.
3	Configuration	CONFIG_ORC_INGEST.OLR_CONFIG	TABLE	OpenLogReplicator settings: Oracle connection (ORACLE_SERVER), reader type, start SCN, path mapping, output format, network URI, archive-log read settings, log level and performance tracing.
4	Configuration	CONFIG_ORC_INGEST.TABLE_FILTER_CONFIG	TABLE	Oracle tables to replicate: table name, Oracle owner, primary key columns, all columns, Snowflake column types and active flag. Updated automatically when DDL is applied.
5	Configuration	CONFIG_ORC_INGEST.FSTREAM_CONFIG	TABLE	SFTP transfer settings: host, port, user, remote archive folder, stage location, start file, workers, run mode and poll interval.
6	Logs	LOGS_ORC_INGEST.CDC_APPLY_STATE	HYBRID TABLE	Apply positions and fencing. Row LWM holds the global position confirmed to OpenLogReplicator. Each TBL_<table> row holds that table's position, committed in the same transaction as its MERGE. OWNER_EPOCH prevents two writer instances applying changes at the same time.
7	Logs	LOGS_ORC_INGEST.CDC_BATCH_LOG	TABLE	One row per table batch (OK) and per failed attempt (ERR): event counts by operation, net keys, SCN range, Oracle commit times, and timings (queue wait, build, MERGE, commit, lag).

Sr. No	Layer	Object Name	Object Type	Description
8	Logs	LOGS_ORC_INGEST.CDC_SERVICE_LOG	TABLE	Service events (STARTUP, CONFIRM, RECONNECT, DDL, FATAL), with full JSON detail.
9	Logs	LOGS_ORC_INGEST.CDC_AUDIT_DDL	TABLE	DDL captured from Oracle: original and translated statements, with outcome APPLIED, LOGGED_UNAPPLIED or FAILED and the error message.
10	Logs	LOGS_ORC_INGEST.CDC_DEAD_LETTER	TABLE	Change events that couldn't be applied (no primary key, or a value that can't be converted), with raw payload and error, for investigation and reprocessing.
11	Logs	LOGS_ORC_INGEST.FSTREAM_AUDIT	TABLE	One row per archive or redo file copied from the Oracle host to the stage: size, source modified time, stable-wait, download and publish timings, and status.
12	Logs	LOGS_ORC_INGEST.FSTREAM_ERROR_LOG	TABLE	Errors from the SFTP file transfer.
13	Logs	LOGS_ORC_INGEST.OLR_FILE_LOG	TABLE	One row each time OpenLogReplicator processes a redo file: sequence, SCN range, start and finish times, processing time, size and supplemental-logging share.
14	Logs	LOGS_ORC_INGEST.V_CDC_TABLE_STATS_MINUTE	VIEW	Per table, per minute: batches, failed attempts, events by operation, dead-letter events, MERGE times, queue wait and lag.
15	Logs	LOGS_ORC_INGEST.V_CDC_TABLE_SUMMARY	VIEW	Per table, all time: totals, last applied SCN, last batch time and last lag.
16	Logs	LOGS_ORC_INGEST.V_ARCHIVE_FILE_AUDIT	VIEW	One row per redo file, joining transfer, OpenLogReplicator processing and the batches that applied its changes, with latency for each stage.
17	Logs	LOGS_ORC_INGEST.V_ARCHIVE_FILE_SUMMARY	VIEW	Hourly roll-up of V_ARCHIVE_FILE_AUDIT: volume, and average and maximum latency for each stage.

Sr. No	Layer	Object Name	Object Type	Description
18	Logs	LOGS_ORC_INGEST.V_ARCHIVE_FILE_LATENCY	VIEW	Simplified per-file latency. The wall-clock steps (Oracle → stage → OpenLogReplicator → applied) add up to the total, and the apply-work breakdown is shown separately.
19	Landing	LANDING_ORC_INGEST.LOGS	STAGE	Internal stage (server-side encrypted, with a directory table) holding the Oracle archive logs transferred over SFTP. Mounted into olr-container and ec2-ingest-container. Its checkpoints/ path holds OpenLogReplicator checkpoints.
20	Target	ORC_INGEST.<TARGET_TABLE>	TABLE	Replicated Oracle table, one row per primary key, plus the pipeline columns IS_DELETED, DELETED_AT, _CDC_SEQ, SNOWFLAKE_INGESTED_AT and SNOWFLAKE_MODIFIED_AT. Created automatically from TABLE_FILTER_CONFIG if missing.

5.4 Objects Details

5.4.1 Object Details in Snowflake Environment

Sr. No	Object Name	Object Type	Description
1	CORE.REGISTER_MY_SECRET	PROCEDURE	Reference callback that binds the consumer's Oracle credential secret and SFTP PEM key secret.
2	CORE.REGISTER_EAI_CALLBACK	PROCEDURE	Reference callback that binds, unbinds or clears the consumer's external access integration.
3	CORE.GET_CONFIGURATION	PROCEDURE	Returns the reference configuration to Snowflake. For the external access integration, it builds the allowed host:port list from ORACLE_SERVER and the SFTP host and port.
4	CORE.ORC_GET_ORACLE_CREDENTIALS	PROCEDURE	Created by CREATE_SERVICE. Returns the bound Oracle username and password to the containers at runtime.
5	CORE.ORC_GET_SFTP_KEY	PROCEDURE	Created by CREATE_SERVICE. Returns the bound SFTP private key, normalised to PEM format, to ec2-ingest-container.

Sr. No	Object Name	Object Type	Description
6	CORE.CREATE_SERVICE	PROCEDURE	Creates the credential procedures, then creates CORE.ORACLE_CDC_SERVICE on ORC_CDC_POOL, using query warehouse ORC_CDC_WH and the bound external access integration.
7	CORE.GET_SERVICE_STATUS	PROCEDURE	Returns the status of each container in the service.
8	CORE.GET_SERVICE_LOGS(INSTANCE_ID, CONTAINER_NAME)	PROCEDURE	Returns logs for olr-container, qtp-container or ec2-ingest-container.
9	CORE.SUSPEND_SERVICE	PROCEDURE	Suspends the CDC service.
10	CORE.RESUME_SERVICE	PROCEDURE	Resumes the CDC service. Replication continues from the saved positions.
11	CORE.DROP_SERVICE	PROCEDURE	Drops the service and keeps the configuration, logs, positions and replicated data.
12	CORE.GRANT_TABLE_ACCESS	PROCEDURE	Grants SELECT on the tables in ORC_INGEST and LOGS_ORC_INGEST to APP_ROLE. Re-run it when new target tables appear.
13	CORE.VERSION_INIT	PROCEDURE	Version initializer called on upgrade. It updates the running service from the new specification file.
14	CORE.ORACLE_CDC_SERVICE	SERVICE	Multi-container SPCS service hosting olr-container, qtp-container and ec2-ingest-container.
15	APP_ROLE	APPLICATION ROLE	The single application role. It has read/write access to the configuration tables, read access to the log tables and views, use of the service procedures, and CREATE TABLE / USAGE on ORC_INGEST.

5.5 Snowflake Container Services (SPCS) Deployment

This section describes how the Oracle CDC application is deployed on Snowpark Container Services (SPCS) within the Snowflake Native App Framework.

- **Packaging:** the components are built as three container images (olr, table_writer, ec2_ingest) in the application package's image repository (/orc_ingest/landing_orc_ingest/orc_img/). They are deployed with service_spec.yaml.

- **Container images:**
  1. olr-container: OpenLogReplicator, which reads the Oracle redo and archive logs.
  2. qtp-container: the table writer, which applies changes to the target tables.
  3. ec2-ingest-container: the SFTP file streamer, which copies archive logs to the stage.

- **Service instantiation:** CORE.CREATE_SERVICE creates CORE.ORACLE_CDC_SERVICE in the consumer's account. The service runs continuously.

- **Inter-container flow:**
  1. ec2-ingest-container copies archive logs over SFTP onto @LANDING_ORC_INGEST.LOGS.
  2. olr-container reads them from the mounted stage (/snowflake_stage) and connects to Oracle.
  3. It streams change events over TCP (default 127.0.0.1:5555, protobuf protocol) to qtp-container.
  4. qtp-container runs MERGEs on ORC_CDC_WH.

- **Volumes:**
  1. main-stage-volume is @LANDING_ORC_INGEST.LOGS, mounted at /snowflake_stage in the OpenLogReplicator and file-streamer containers.
  2. checkpoint-volume is @LANDING_ORC_INGEST.LOGS/checkpoints, mounted in the OpenLogReplicator and table-writer containers.

- **Isolated compute:** the service runs on an app-owned compute pool inside the Native App's security boundary. The data stays in the consumer's account.

- **Reference-based configuration:** runtime behaviour comes from the configuration tables in CONFIG_ORC_INGEST plus three bound references: the Oracle secret, the PEM secret and the external access integration.

5.5.1 Compute Pool Configuration

- The application creates and owns a dedicated compute pool. This requires the CREATE COMPUTE POOL privilege, approved during installation.
- It has a fixed one-node size: MIN_NODES = MAX_NODES = 1.
- The three containers together request 3.2 vCPU and 3.5 GiB, so the node type must give at least that.

Sr. No	Object Name	Object Type	Description
1	ORC_CDC_POOL	Compute Pool	App-created pool: INSTANCE_FAMILY = CPU_X64_M (6 vCPU, 28 GiB), MIN_NODES = 1, MAX_NODES = 1, AUTO_RESUME = TRUE, AUTO_SUSPEND_SECS = 60. Runs CORE.ORACLE_CDC_SERVICE.
2	ORC_CDC_WH	Warehouse	App-created warehouse (privilege CREATE WAREHOUSE): Generation 2, SMALL, AUTO_SUSPEND = 60, AUTO_RESUME = TRUE, initially suspended. Query warehouse for the service's MERGE, position and log statements.

5.5.2 Snowflake Service Details

The Oracle CDC logic runs as an SPCS service inside the Native App instance.

Internal service:

- ec2-ingest-container transfers the archive logs.
- olr-container captures the changes.
- qtp-container applies them as one MERGE per table batch. Each table's position is committed in the same transaction.

Infrastructure binding: the service runs on the app-owned ORC_CDC_POOL with query warehouse ORC_CDC_WH. The consumer doesn't bind a compute pool or a warehouse.

Secure connectivity:
Oracle (port 1521) and SFTP (port 22) are reached through the bound external access integration. Credentials come from the bound secrets through CORE.ORC_GET_ORACLE_CREDENTIALS and CORE.ORC_GET_SFTP_KEY when the containers start.

Resources requested by each container:

Container	CPU request	CPU limit	Memory request	Memory limit
olr-container	1.0	2.0	1 GiB	3 GiB
qtp-container	2.0	4.0	2 GiB	6 GiB
ec2-ingest-container	0.2	0.3	512 MiB	2 GiB

Monitoring: SPCS platform metrics (system, system_limits) are enabled in the service spec.

Lifecycle management: the service is managed through CORE.CREATE_SERVICE, SUSPEND_SERVICE, RESUME_SERVICE, DROP_SERVICE, GET_SERVICE_STATUS, GET_SERVICE_LOGS and VERSION_INIT. It has no public endpoints.

Sr. No	Service Name	Description
1	CORE.ORACLE_CDC_SERVICE	Continuous multi-container service with three containers: olr-container (OpenLogReplicator), qtp-container (table writer) and ec2-ingest-container (SFTP archive-log streamer). It captures Oracle changes and applies them to ORC_INGEST target tables.

5.6 Oracle to Snowflake Data Type Mapping

The following mapping is applied during extraction and landing. Types not listed are cast to VARCHAR.

Oracle Type	Snowflake Type	Notes
NUMBER(p,0), INTEGER	NUMBER(p,0)	Integer values; precision depends on the source definition.
NUMBER(p,s), NUMERIC, DECIMAL	NUMBER(p,s)	Snowflake supports precision up to 38 digits.
BINARY_FLOAT, BINARY_DOUBLE	FLOAT	Floating-point values.
CHAR(n), NCHAR(n), VARCHAR2(n), NVARCHAR2(n)	VARCHAR	Character data; length and character semantics should be considered.
DATE	TIMESTAMP_NTZ	Oracle DATE includes both date and time to seconds.

Oracle Type	Snowflake Type	Notes
TIMESTAMP(p)	TIMESTAMP_NTZ	Fractional-second precision depends on the source definition.
TIMESTAMP(p) WITH TIME ZONE	TIMESTAMP_TZ	Time zone information is represented in the target type.
TIMESTAMP(p) WITH LOCAL TIME ZONE	TIMESTAMP_TZ	Conversion depends on Oracle session time-zone semantics and the CDC mapping implementation.
CLOB, NCLOB, LONG	VARCHAR	Text data; practical size and conversion limits depend on the ingestion implementation.
BLOB, RAW(n), LONG RAW	BINARY	Binary data; conversion depends on the source type and ingestion implementation.
JSON	VARIANT	Semi-structured representation, where supported by the ingestion logic.
XMLTYPE	VARCHAR / VARIANT	Depends on the configured conversion strategy.
INTERVAL YEAR TO MONTH, INTERVAL DAY TO SECOND	VARCHAR	Text representation if no native interval conversion is implemented.
ROWID, UROWID	VARCHAR	Character representation.
BFILE, OBJECT, VARRAY, nested table types	Implementation-dependent	Requires explicit handling based on the source definition and CDC serialization logic.

5.7 Extraction Logic

- **Initial Load:** Oracle table data is captured through the configured CDC process, with the OLR publisher reading Oracle redo and archive logs and publishing change events to the consumer.
- **Incremental CDC:** The consumer processes change events from the configured OLR start position and uses checkpoint and audit records to track processing progress and support recovery.
- **Bounded Batching:** Change events are accumulated into batches subject to configured row-count and byte-size limits to control memory usage and processing overhead.
- **Transaction-Aware Processing:** The consumer processes change events with transaction boundaries in mind before applying changes to Snowflake target tables.
- **Checkpoint and Audit:** Processing progress and CDC audit information are recorded in Snowflake to support recovery, traceability, and monitoring.

5.8 Representative Table Structures

The following column layouts illustrate the information held in the Oracle CDC configuration and logging tables. They are representative and must be aligned with the DDL shipped in the deployed application version.

5.8.1 CONFIG_ORC_INGEST.CDC_JOB_REGISTRY

Column (representative)	Purpose
JOB_ID	Unique identifier of the CDC job
JOB_NAME	Name of the CDC job
DESCRIPTION	Description of the job
IS_ACTIVE	Enables or disables the job
CREATED_AT / UPDATED_AT	When the job record was created / last updated

5.8.2 CONFIG_ORC_INGEST.CDC_CONFIG

Column (representative)	Purpose
JOB_ID	Job associated with the CDC configuration
SNOWFLAKE_WAREHOUSE / SNOWFLAKE_DATABASE	Warehouse and database used by the consumer (NULL = service defaults)
APP_SCHEMA / AUDIT_SCHEMA	Schemas for target data and audit records
OLR_HOST / OLR_PORT / OLR_DATABASE	Connection to the OLR publisher and its source identifier
OLR_START	Starting position for CDC processing (`now` or `scn:<n>`)
CHECKPOINT_FILE	Path of the consumer checkpoint file
MAX_QUEUE_COUNT / MAX_QUEUE_BYTES	Limits for the event queue
APPLY_MAX_EVENTS / TABLE_INBOX_MAX	Maximum events per MERGE; queued messages per table before backpressure
APPLY_SESSIONS	Number of parallel apply workers (0 = warehouse concurrency − 2)
APPLY_MODE	How a batch is written: MERGE, SPLIT_MERGE or INSERT_MERGE
CONFIRM_INTERVAL_MS	How often the global position is saved and confirmed
RECONNECT_MIN_MS / RECONNECT_MAX_MS	Reconnect backoff to OLR
SNOWFLAKE_ALERT_AFTER_S / LOG_SUMMARY_INTERVAL_S	FATAL alert threshold; SUMMARY log interval
IS_ACTIVE / UPDATED_AT	Enables the configuration; last update time

5.8.3 CONFIG_ORC_INGEST.OLR_CONFIG

Column (representative)	Purpose
JOB_ID	Job associated with the OpenLogReplicator configuration
ORACLE_SERVER	Oracle connection string (`//host:port/service`)
SOURCE_ALIAS / SOURCE_NAME / SOURCE_FLAGS	OLR source definition
READER_TYPE / ARCH	Redo reader type and archive log mode
START_SCN	Starting SCN for OLR (0 = default)
PATH_MAPPING	Maps Oracle log paths to the stage mount
FORMAT_TYPE / TARGET_ALIAS / NETWORK_URI	Output format and network target for the consumer
POLL_INTERVAL_US / ARCH_READ_SLEEP_US / ARCH_READ_TRIES	Polling and archive read retry settings
LOG_LEVEL / TRACE_PERFORMANCE	OLR log level and per-file performance tracing
IS_ACTIVE / UPDATED_AT	Enables the configuration; last update time

5.8.4 CONFIG_ORC_INGEST.TABLE_FILTER_CONFIG

Column (representative)	Purpose
CONFIG_ID	Unique identifier of the table entry
JOB_ID	Job the table belongs to
TABLE_NAME / ORACLE_OWNER	Oracle table and owner to replicate
PK_COLUMNS	Primary key columns
ALL_COLUMNS	All columns to replicate
COLUMN_TYPES	Snowflake type per column, used to create the target table
IS_ACTIVE	Enables or pauses replication of the table
CREATED_AT	When the entry was created

5.8.5 CONFIG_ORC_INGEST.FSTREAM_CONFIG

Column (representative)	Purpose
JOB_ID	Job associated with the file transfer configuration
SFTP_HOST / SFTP_PORT / SFTP_USER	SSH connection to the Oracle host
REMOTE_FOLDER	Oracle archive log directory
STAGE_MOUNT_DIR / STAGE_LOCATION	Target location on the stage
SFTP_TIMEOUT / MAX_WORKERS	Transfer timeout and parallel transfers
RUN_MODE / POLL_INTERVAL_SECONDS	CONTINUOUS or BATCH; polling interval

START_FROM_FILE	First archive file to copy
SNOWFLAKE_SCHEMA / AUDIT_TABLE	Where file transfer audit rows are written
IS_ACTIVE / CREATED_AT / UPDATED_AT	Enables the configuration; timestamps

5.8.6 LOGS_ORC_INGEST.CDC_APPLY_STATE (hybrid table)

Column (representative)	Purpose
NAME	`LWM` (global position) or `TBL_<table>` (per-table position)
OWNER_EPOCH	Fencing token that prevents two service instances writing at once
LWM_SCN / LWM_IDX	Last applied position; everything up to it is applied
UPDATED_AT	When the position was last saved

5.8.7 LOGS_ORC_INGEST.CDC_BATCH_LOG

Column (representative)	Purpose
BATCH_ID / TABLE_NAME	Batch and table applied
STATUS / ATTEMPT / ERROR_MESSAGE	OK or ERR, attempt number, error text
MESSAGES / EVENTS	Source transactions and row changes in the batch
INSERT_EVENTS / UPDATE_EVENTS / DELETE_EVENTS / DLQ_EVENTS	Change counts by type; rows sent to dead letter
NET_PKS / UPSERTS / SOFT_DELETES / PK_CHANGES	Net result after per-key netting
ROWS_INSERTED / ROWS_UPDATED	Rows reported by the MERGE
START_SCN / END_SCN	SCN range of the batch
MIN_SOURCE_TS / MAX_SOURCE_TS	Oracle commit times of the oldest / newest change
QUEUE_WAIT_MS / BUILD_MS / MERGE_MS / TOTAL_MS / LAG_MS	Timings and end-to-end lag
INSERT_ROWS / OTHER_ROWS / INSERT_MERGE_MS / OTHER_MERGE_MS	Split timings (SPLIT_MERGE / INSERT_MERGE only)
LOGGED_AT	When the row was logged

5.8.8 LOGS_ORC_INGEST.CDC_SERVICE_LOG

Column (representative)	Purpose
EVENT_TS / EVENT_TYPE	Event time and type (STARTUP, SUMMARY, CONFIRM, RECONNECT, DDL, FATAL)
TABLE_NAME / STATUS	Table and status, when applicable
OWNER_EPOCH	Service instance that logged the event
DETAIL	Full JSON record of the event
LOGGED_AT	When the row was logged

5.8.9 LOGS_ORC_INGEST.CDC_AUDIT_DDL

Column (representative)	Purpose
SCN / C_SCN / C_IDX	Oracle position of the DDL
TABLE_NAME	Table affected
ORIGINAL_DDL / TRANSLATED_DDL	Oracle DDL and the Snowflake statement derived from it
ACTION_STATUS	APPLIED, LOGGED_UNAPPLIED or FAILED
ERROR_MESSAGE / CREATED_AT	Error text; when it was logged

5.8.10 LOGS_ORC_INGEST.CDC_DEAD_LETTER

Column (representative)	Purpose
EVENT_ID	Unique identifier of the rejected change
C_SCN / C_IDX	Oracle position of the change
TABLE_NAME	Target table
RAW_PAYLOAD	Original change event
ERROR_MSG / CREATED_AT	Why it was rejected; when it was logged

5.8.11 LOGS_ORC_INGEST.FSTREAM_AUDIT

Column (representative)	Purpose
JOB_ID / RUN_ID	Job and run that copied the file
FILE_NAME / REMOTE_PATH / STAGE_PATH	Archive file and its source and target paths
FILE_SIZE_BYTES	File size
SOURCE_MODIFIED_AT / DETECTED_AT	When Oracle finished the file; when it was first seen
START_TIME / END_TIME	Transfer window
STATUS	Processing, Success or Failure
STABLE_WAIT_MS / DOWNLOAD_MS / PUBLISH_MS	Time spent in each transfer step
DURATION_SECONDS / TRANSFER_SPEED_MBPS	Total transfer time and speed
ERROR_MESSAGE / PROCESSED_AT	Error text; when the row was written

5.8.12 LOGS_ORC_INGEST.FSTREAM_ERROR_LOG

Column (representative)	Purpose
JOB_ID / RUN_ID	Job and run
FILE_NAME	File that failed
START_TIME	When the transfer started
ERROR_MESSAGE / STATUS	Error text and status

ERROR_LOG_KEY / CREATED_AT	Error record key; when it was logged

5.8.13 LOGS_ORC_INGEST.OLR_FILE_LOG

Column (representative)	Purpose
FILE_NAME / PATH	Redo file processed by OpenLogReplicator
GROUP_NO / SEQUENCE	0 = archived log, > 0 = online log group; log sequence
FIRST_SCN / NEXT_SCN	SCN range of the file
STARTED_AT / FINISHED_AT / FINISH_SOURCE	Processing window and how the end was detected
OLR_MS	OLR's own processing time
REDO_SIZE_MB / READ_SIZE_MB / SPEED_MBS	File size and read speed
SUPP_LOG_BYTES / SUPP_LOG_PCT	Share of the file used by supplemental logging
LOGGED_AT	When the row was logged

5.8.14 LOGS_ORC_INGEST monitoring views

View	Purpose
V_CDC_TABLE_STATS_MINUTE	Per table, per minute: batches, events, MERGE time, queue wait and lag
V_CDC_TABLE_SUMMARY	Per table: totals, last applied SCN, last batch time and lag
V_ARCHIVE_FILE_AUDIT	Per redo file: full detail of transfer, OLR and apply
V_ARCHIVE_FILE_LATENCY	Per redo file: time spent in each step, from Oracle to applied
V_ARCHIVE_FILE_SUMMARY	Per hour: files, volume, and average / maximum time per step

5.8.15 ORC_INGEST replicated tables

Column (representative)	Purpose
Source columns	Columns from `TABLE_FILTER_CONFIG.ALL_COLUMNS`
IS_DELETED / DELETED_AT	Soft delete flag and Oracle commit time of the delete
_CDC_SEQ	Oracle position of the last change applied to the row
SNOWFLAKE_INGESTED_AT / SNOWFLAKE_MODIFIED_AT	When the row was first written / last changed

5.9 Application Roles and Privileges

Role / Privilege	Type	Purpose
APP_ADMIN	Application role	Administer the app: configure jobs and tables, start and suspend services, view logs
Read-only application role	Application role	View curated tables and logging tables without administrative rights

Role / Privilege	Type	Purpose
CREATE COMPUTE POOL	Account privilege	Allows the app to manage its own compute pool
BIND SERVICE ENDPOINT	Account privilege	Allows service endpoints required by the app
USAGE on EAI / READ on Secret	Reference	Granted through Reference Binding so the service can reach and authenticate Oracle
USAGE on Warehouse	Reference	Allows merge and transformation statements to run

6 Oracle Source Preparation

This section is intended for the Oracle DBA. It lists the steps required to prepare a Oracle database as an ingestion source.

6.1 Supported Environments

Environment	Notes
Self-managed Oracle Database	Oracle host must provide access to online redo and archived redo logs.
Oracle Database on AWS EC2	Security groups must allow Oracle connectivity and SSH/SFTP access.
Oracle Database on OCI	Network security lists and NSGs must allow the required connectivity.
Oracle Exadata	Network access and redo/archive log access must be configured.
Amazon RDS for Oracle	Validate archive-log extraction and access restrictions; direct filesystem access is generally restricted.

6.2 Create a Dedicated CDC User

Create a dedicated role for Pi-Ingest with the minimum privileges required. Run the following as a Oracle administrator:

```
CREATE USER PI_INGEST_USER
IDENTIFIED BY "<strong_password>";

GRANT CREATE SESSION TO PI_INGEST_USER;

-- Dictionary access required by OLR
GRANT SELECT, FLASHBACK ON SYS.CCOL$ TO PI_INGEST_USER;
GRANT SELECT, FLASHBACK ON SYS.CDEF$ TO PI_INGEST_USER;
GRANT SELECT, FLASHBACK ON SYS.COL$ TO PI_INGEST_USER;
GRANT SELECT, FLASHBACK ON SYS.DEFERRED_STG$ TO PI_INGEST_USER;
GRANT SELECT, FLASHBACK ON SYS.ECOL$ TO PI_INGEST_USER;
GRANT SELECT, FLASHBACK ON SYS.LOB$ TO PI_INGEST_USER;
GRANT SELECT, FLASHBACK ON SYS.LOBCOMPPART$ TO PI_INGEST_USER;
GRANT SELECT, FLASHBACK ON SYS.LOBFRAG$ TO PI_INGEST_USER;
GRANT SELECT, FLASHBACK ON SYS.OBJ$ TO PI_INGEST_USER;
GRANT SELECT, FLASHBACK ON SYS.TAB$ TO PI_INGEST_USER;
GRANT SELECT, FLASHBACK ON SYS.TABCOMPART$ TO PI_INGEST_USER;
GRANT SELECT, FLASHBACK ON SYS.TABPART$ TO PI_INGEST_USER;
GRANT SELECT, FLASHBACK ON SYS.TABSUBPART$ TO PI_INGEST_USER;
GRANT SELECT, FLASHBACK ON SYS.TS$ TO PI_INGEST_USER;
GRANT SELECT, FLASHBACK ON SYS.USER$ TO PI_INGEST_USER;
GRANT SELECT ON SYS.V_$ARCHIVED_LOG TO PI_INGEST_USER;
GRANT SELECT ON SYS.V_$DATABASE TO PI_INGEST_USER;
GRANT SELECT ON SYS.V_$DATABASE_INCARNATION TO PI_INGEST_USER;
GRANT SELECT ON SYS.V_$LOG TO PI_INGEST_USER;
GRANT SELECT ON SYS.V_$LOGFILE TO PI_INGEST_USER;
GRANT SELECT ON SYS.V_$PARAMETER TO PI_INGEST_USER;
GRANT SELECT ON SYS.V_$PDBS TO PI_INGEST_USER;
GRANT SELECT ON SYS.V_$STANDBY_LOG TO PI_INGEST_USER;
GRANT SELECT ON SYS.V_$TRANSPORTABLE_PLATFORM TO PI_INGEST_USER;
GRANT SELECT, FLASHBACK ON XDB.XDB$TTSET TO PI_INGEST_USER;


-- dynamic grants
```

```
DECLARE
    CURSOR C1 IS SELECT TOKSUF FROM XDB.XDB$TTSET;
    CMD VARCHAR2(2000);
BEGIN
    FOR C IN C1 LOOP
        CMD := 'GRANT SELECT, FLASHBACK ON XDB.X$NM' || C.TOKSUF || ' TO PI_INGEST_USER';
        EXECUTE IMMEDIATE CMD;
        CMD := 'GRANT SELECT, FLASHBACK ON XDB.X$QN' || C.TOKSUF || ' TO PI_INGEST_USER';
        EXECUTE IMMEDIATE CMD;
        CMD := 'GRANT SELECT, FLASHBACK ON XDB.X$PT' || C.TOKSUF || ' TO PI_INGEST_USER';
        EXECUTE IMMEDIATE CMD;
    END LOOP;
END;
/

-- Access to captured application tables
GRANT SELECT ON <APP_SCHEMA>.<TABLE_1> TO PI_INGEST_USER;
GRANT SELECT ON <APP_SCHEMA>.<TABLE_2> TO PI_INGEST_USER;

-- verify all grants have been applied :

SELECT GRANT_STATUS, COUNT(*) AS TOTAL
FROM (
    WITH required_grants AS (
    -- 15 SYS dictionary objects: SELECT + FLASHBACK
    SELECT 'SYS' OWNER, COLUMN_VALUE TABLE_NAME, 'SELECT' PRIVILEGE
    FROM TABLE(SYS.ODCIVARCHAR2LIST(
        'CCOL$', 'CDEF$', 'COL$', 'DEFERRED_STG$', 'ECOL$',
        'LOB$', 'LOBCOMPPART$', 'LOBFRAG$', 'OBJ$', 'TAB$',
        'TABCOMPART$', 'TABPART$', 'TABSUBPART$', 'TS$', 'USER$'
    ))
    UNION ALL
    SELECT 'SYS', COLUMN_VALUE, 'FLASHBACK'
    FROM TABLE(SYS.ODCIVARCHAR2LIST(
        'CCOL$', 'CDEF$', 'COL$', 'DEFERRED_STG$', 'ECOL$',
        'LOB$', 'LOBCOMPPART$', 'LOBFRAG$', 'OBJ$', 'TAB$',
        'TABCOMPART$', 'TABPART$', 'TABSUBPART$', 'TS$', 'USER$'
    ))

    UNION ALL

    -- 9 SYS V$ objects: SELECT
    SELECT 'SYS', COLUMN_VALUE, 'SELECT'
    FROM TABLE(SYS.ODCIVARCHAR2LIST(
        'V_$ARCHIVED_LOG', 'V_$DATABASE',
        'V_$DATABASE_INCARNATION', 'V_$LOG', 'V_$LOGFILE',
        'V_$PARAMETER', 'V_$PDBS', 'V_$STANDBY_LOG',
        'V_$TRANSPORTABLE_PLATFORM'
    ))

    UNION ALL

    -- XDB$TTSET: SELECT + FLASHBACK
    SELECT 'XDB', 'XDB$TTSET', 'SELECT' FROM DUAL
    UNION ALL
    SELECT 'XDB', 'XDB$TTSET', 'FLASHBACK' FROM DUAL

    UNION ALL
```

```
    -- Dynamic XDB objects: SELECT + FLASHBACK
    SELECT 'XDB', TABLE_NAME, PRIVILEGE
    FROM (
        SELECT 'X$NM' || TOKSUF TABLE_NAME FROM XDB.XDB$TTSET
        UNION ALL
        SELECT 'X$QN' || TOKSUF FROM XDB.XDB$TTSET
        UNION ALL
        SELECT 'X$PT' || TOKSUF FROM XDB.XDB$TTSET
    )
    CROSS JOIN (
        SELECT 'SELECT' PRIVILEGE FROM DUAL
        UNION ALL
        SELECT 'FLASHBACK' FROM DUAL
    )
)
SELECT
    R.OWNER,
    R.TABLE_NAME,
    R.PRIVILEGE,
    CASE
        WHEN P.GRANTEE IS NOT NULL THEN 'GRANTED'
        ELSE 'MISSING'
    END AS GRANT_STATUS
FROM required_grants R
LEFT JOIN DBA_TAB_PRIVS P
    ON P.OWNER = R.OWNER
   AND P.TABLE_NAME = R.TABLE_NAME
   AND P.GRANTEE = 'PI_INGEST_USER'
   AND P.PRIVILEGE = R.PRIVILEGE
)
GROUP BY GRANT_STATUS;

-- expected output:
GRANT_S      TOTAL
------- ----------
GRANTED         41 + 6N

where N = SELECT COUNT(*) AS N FROM XDB.XDB$TTSET;
```

6.3 Preparing Source Database

```
-- Enable database-level supplemental logging
ALTER DATABASE ADD SUPPLEMENTAL LOG DATA;

-- Enable primary-key supplemental logging
ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (PRIMARY KEY) COLUMNS;

-- Enable unique-key supplemental logging
ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (UNIQUE) COLUMNS;

-- Enable foreign-key supplemental logging
ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (FOREIGN KEY) COLUMNS;

-- Enable all-column supplemental logging
 ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (ALL) COLUMNS;
```

```
-- Verify supplemental logging configuration
SELECT
        SUPPLEMENTAL_LOG_DATA_MIN,
        SUPPLEMENTAL_LOG_DATA_PK,
        SUPPLEMENTAL_LOG_DATA_UI,
        SUPPLEMENTAL_LOG_DATA_FK,
        SUPPLEMENTAL_LOG_DATA_ALL
FROM
        V$DATABASE;
```

6.4 Network and Authentication

- **Oracle Listener:** Ensure the Oracle listener is configured to accept remote connections on the database port (default 1521).
- **Firewall / Security Group:** Allow inbound traffic on port 1521 from approved Snowflake ingestion source addresses for direct database connectivity.
- **SFTP:** If archived redo logs are transferred from the Oracle host to Snowflake, allow SFTP connectivity on port 22 from the archive-ingestion container to the Oracle host.
- **SSL / Encryption:** Use Oracle TCPS supported, or an approved encrypted network path. For archive-log transfer, use SSH/SFTP.
- **Service Name:** Ensure the correct Oracle service name or PDB service name is configured for the connection.
- **Authentication**: Store the dedicated CDC user's credentials securely in the Snowflake secret/configuration mechanism.

6.4 Recommended Server Settings

Setting	Recommendation	Reason
SESSIONS / PROCESSES	Leave sufficient headroom for ingestion and application connections.	Prevents connection exhaustion.
LOG_MODE	Enable ARCHIVELOG mode.	Required for archived redo log-based CDC.
Supplemental logging	Enable database-level supplemental logging.	Provides additional information in redo records.
Primary-key supplemental logging	Enable for all replicated tables.	Supports correct identification of rows during UPDATE and DELETE operations.
Archive destination	Maintain sufficient disk capacity and monitor usage.	Prevents archive failures due to insufficient storage.
Archive retention	Retain logs until OLR has successfully processed and confirmed them.	Prevents missing changes during ingestion delays.
Redo log size	Size redo log groups according to transaction volume.	Helps avoid excessive log switches.
Primary keys	Ensure replicated tables have primary keys.	Supports reliable UPDATE and DELETE replication.

Setting	Recommendation	Reason
RAC redo threads	Configure ingestion to account for all redo threads, if using RAC.	Ensures changes from all instances are captured.

6.5 Source Readiness Checklist

#	Check	Owner
1	Dedicated CDC user created with required OLR dictionary privileges	Oracle DBA
2	SELECT privileges granted on all replicated application tables	Oracle DBA
3	Network connectivity to Oracle listener (port 1521) verified	Network / DBA
4	SFTP connectivity (port 22) configured for archive-log transfer, if applicable	Network / DBA
5	Secure authentication and credential storage configured	Security / DBA
6	ARCHIVELOG mode enabled	Oracle DBA
7	Database-level supplemental logging enabled	Oracle DBA
8	Primary-key supplemental logging enabled for replicated tables	Oracle DBA
9	Primary keys identified for all replicated tables	Data Engineer
10	Archive destination capacity and retention policy confirmed	Oracle DBA
11	Redo log sizing and log-switch frequency reviewed	Oracle DBA
12	RAC redo threads and archive sequences accounted for, if applicable	Oracle DBA
13	Peak transaction volume and expected CDC latency agreed	DBA / Data Engineer
14	Maximum ingestion concurrency agreed	DBA / Data Engineer
15	Credential rotation process agreed	Security

7 Snowflake Setup and Configuration

This section is intended for the consumer Snowflake administrator. The statements below illustrate how consumer-owned objects are created before they are bound to the application.

7.1 Create the Network Rule

```
CREATE OR REPLACE NETWORK RULE oracle_host_rule
  MODE = EGRESS
  TYPE = HOST_PORT
  VALUE_LIST = ('<oracle_host>:1521', '<sftp_host>:22');
```

7.2 Create the Secret

```
USE DATABASE <my_database>;
USE SCHEMA <my_schema>;

-- Oracle database user for OpenLogReplicator
CREATE OR REPLACE SECRET oracle_credentials
  TYPE = PASSWORD
  USERNAME = '<oracle_user>'
  PASSWORD = '<oracle_password>';

-- SSH private key (.pem), including the BEGIN / END lines
CREATE OR REPLACE SECRET sftp_pem_key
  TYPE = GENERIC_STRING
  SECRET_STRING = '<contents of the .pem file>';
```

7.3 Create the External Access Integration

```
CREATE OR REPLACE EXTERNAL ACCESS INTEGRATION oracle_cdc_eai
  ALLOWED_NETWORK_RULES = (oracle_host_rule)
  ALLOWED_AUTHENTICATION_SECRETS = (oracle_credentials, sftp_pem_key)
  ENABLED = TRUE;
```

7.4 Application Privileges and Reference Binding

The application asks for two account privileges, which the consumer approves during installation:

Privilege	Purpose
`CREATE COMPUTE POOL`	The app creates its own compute pool `ORC_CDC_POOL` (CPU_X64_M, 1 node)
`CREATE WAREHOUSE`	The app creates its own warehouse `ORC_CDC_WH` (Gen2, SMALL, auto-suspend 60 s)

Note: *Binding of the Secret, External Access Integration, Compute Pool, and Warehouse to the application references is completed through the application Settings UI or the reference procedures provided by the app.*

7.5 Configure the Source

```
USE APPLICATION <app_name>;

UPDATE config_orc_ingest.olr_config
SET ORACLE_SERVER = '//<oracle_host>:1521/<service_name>'
WHERE JOB_ID = 'ORACLE_CDC_JOB_01';

UPDATE config_orc_ingest.fstream_config
SET SFTP_HOST     = '<sftp_host>',
    SFTP_USER     = '<ssh_user>',
    REMOTE_FOLDER = '<oracle_archive_dir>'
```

```
WHERE JOB_ID = 'ORACLE_CDC_JOB_01';
```

7.6 Register Tables for Replication

Add one row to CONFIG_ORC_INGEST.TABLE_FILTER_CONFIG for each Oracle table. Each row gives the table name, the Oracle owner, the primary key columns, all the columns, and the Snowflake type of each column.

Example:

```
INSERT INTO config_orc_ingest.table_filter_config
    (JOB_ID, TABLE_NAME, ORACLE_OWNER, PK_COLUMNS, ALL_COLUMNS, COLUMN_TYPES)
SELECT 'ORACLE_CDC_JOB_01', 'CUSTOMERS', 'SALES',
       PARSE_JSON('["ID"]'),
       PARSE_JSON('["ID", "NAME", "CREATED_AT"]'),
       PARSE_JSON('{"ID": "NUMBER(10,0)", "NAME": "VARCHAR(100)", "CREATED_AT":
"TIMESTAMP_NTZ(6)"}');
```

- **Names:** write table and column names in capital letters, the way Oracle stores them.
- **Target table:** when the service starts, it creates ORC_INGEST.<TABLE_NAME> if it doesn't exist yet. It adds the pipeline columns IS_DELETED, DELETED_AT, _CDC_SEQ, SNOWFLAKE_INGESTED_AT and SNOWFLAKE_MODIFIED_AT.
- **CDC only:** the connector replicates changes from the start position onward. There is no full or initial load, and there are no watermark columns. Rows that existed before the start position must be loaded separately.
- **Pausing a table**: set IS_ACTIVE = FALSE on its row to stop replicating it. The target table and its logs are kept.
- **Supplemental logging:** it must be enabled in Oracle for every registered table, at least on the primary key. (ALL) COLUMNS is recommended.

8 Operations and Runbook

The connector runs continuously as one SPCS service, CORE.ORACLE_CDC_SERVICE. There are no scheduled runs. Health is judged by how far behind replication is, whether failed batches are piling up, and whether the dead-letter table is growing. All timestamps in the log tables are UTC.

8.1 Daily Operational Checks

Check	How	Expected result
Service status	`CALL core.get_service_status()`	`olr-container`, `qtp-container` and `ec2-ingest-container` all `READY`
Replication lag	`LAST_LAG_MS` and `LAST_BATCH_AT` in `LOGS_ORC_INGEST.V_CDC_TABLE_SUMMARY`	Lag within the agreed freshness target
Positions moving	`UPDATED_AT` in `LOGS_ORC_INGEST.CDC_APPLY_STATE`	The `LWM` row updated within the last few seconds
Failed batches	`STATUS = 'ERR'` in `LOGS_ORC_INGEST.CDC_BATCH_LOG`, last 24 h	None, or only a few that were retried successfully
Dead-letter rows	`LOGS_ORC_INGEST.CDC_DEAD_LETTER`, last 24 h	No new rows
Service events	`EVENT_TYPE IN ('FATAL','RECONNECT')` in `LOGS_ORC_INGEST.CDC_SERVICE_LOG`	No FATAL; RECONNECT only now and then
DDL changes	`ACTION_STATUS <> 'APPLIED'` in `LOGS_ORC_INGEST.CDC_AUDIT_DDL`	Each one reviewed (e.g. RENAME TABLE needs manual action)
Archive log transfer	`LOGS_ORC_INGEST.FSTREAM_AUDIT` / `FSTREAM_ERROR_LOG`	New files arriving as `Success`; no new errors
End-to-end latency	`LOGS_ORC_INGEST.V_ARCHIVE_FILE_LATENCY`	`TOTAL_LATENCY_MS` stable; no step growing
Compute pool	`SHOW COMPUTE POOLS LIKE 'ORC_CDC_POOL'`	`ACTIVE`, 1 node

Reconciliation: there is no source-vs-loaded row count, because this is change capture, not table loads. To check that the data matches, compare COUNT(*) WHERE NOT IS_DELETED in the target table with COUNT(*) in Oracle at the same time.

8.2 Routine Operations

Operation	Procedure
Add a table	Enable supplemental logging on the table in Oracle. Insert a row into `CONFIG_ORC_INGEST.TABLE_FILTER_CONFIG`, then restart the service (suspend + resume). The target table is created automatically. Only changes from then on are captured; existing rows need a separate one-off load.
Pause a table	Set `IS_ACTIVE = FALSE` on its `TABLE_FILTER_CONFIG` row and restart the service. The target table and its logs are kept.
Change settings	Update `CDC_CONFIG`, `OLR_CONFIG` or `FSTREAM_CONFIG`, then restart the service.
Rotate credentials	Update the Snowflake secret (`ALTER SECRET oracle_credentials SET PASSWORD = ...`, or the PEM secret), then restart the service. The secrets are read when the containers start.
Suspend replication	`CALL core.suspend_service()`. On resume, replication continues from the saved position, as long as Oracle still has the archive logs.
Resume replication	`CALL core.resume_service()`
Recreate the service	`CALL core.drop_service()` then `CALL core.create_service()`. Configuration, logs, positions and replicated data are kept.
Grant data access	`CALL core.grant_table_access()` after new tables appear. Then `GRANT APPLICATION ROLE <app_name>.app_role TO ROLE <role>`.
Reprocess dead-letter rows	Fix the cause, then reapply the rows from `RAW_PAYLOAD` by hand, or re-sync the affected keys from Oracle.
Upgrade the app	Install the new version or patch from the listing. `CORE.VERSION_INIT` updates the service specification. Then check the reference bindings and `get_service_status()`.

8.3 Useful Diagnostic Queries

```
USE APPLICATION <app_name>;

-- Per-table health: last batch, lag, failures, dead letters
SELECT TABLE_NAME, LAST_BATCH_AT, LAST_LAG_MS, BATCHES, FAILED_ATTEMPTS, DLQ_EVENTS
FROM logs_orc_ingest.v_cdc_table_summary
ORDER BY LAST_BATCH_AT;

-- Positions (global LWM and per table)
SELECT NAME, LWM_SCN, LWM_IDX, UPDATED_AT
FROM logs_orc_ingest.cdc_apply_state
ORDER BY NAME;

-- Failed batch attempts, last 24 h
SELECT FINISHED_AT, TABLE_NAME, ATTEMPT, ERROR_MESSAGE
FROM logs_orc_ingest.cdc_batch_log
WHERE STATUS = 'ERR' AND LOGGED_AT > DATEADD(hour, -24, SYSDATE())
ORDER BY LOGGED_AT DESC;

-- Dead-letter rows
SELECT CREATED_AT, TABLE_NAME, ERROR_MSG, RAW_PAYLOAD
FROM logs_orc_ingest.cdc_dead_letter
```

```
ORDER BY CREATED_AT DESC LIMIT 50;

-- Service events (startup, reconnects, fatal errors)
SELECT EVENT_TS, EVENT_TYPE, TABLE_NAME, STATUS, DETAIL
FROM logs_orc_ingest.cdc_service_log
WHERE EVENT_TYPE <> 'CONFIRM'
ORDER BY EVENT_TS DESC LIMIT 50;

-- DDL captured from Oracle that was not applied
SELECT CREATED_AT, TABLE_NAME, ACTION_STATUS, ORIGINAL_DDL, ERROR_MESSAGE
FROM logs_orc_ingest.cdc_audit_ddl
WHERE ACTION_STATUS <> 'APPLIED'
ORDER BY CREATED_AT DESC;

-- Archive log transfers and per-file latency
SELECT * FROM logs_orc_ingest.fstream_error_log ORDER BY CREATED_AT DESC LIMIT 20;
SELECT * FROM logs_orc_ingest.v_archive_file_latency ORDER BY OLR_STARTED_AT DESC LIMIT 50;

-- Service status and container logs (instance 0)
CALL core.get_service_status();
CALL core.get_service_logs(0, 'olr-container');
CALL core.get_service_logs(0, 'qtp-container');
CALL core.get_service_logs(0, 'ec2-ingest-container');

SHOW COMPUTE POOLS LIKE 'ORC_CDC_POOL';
```

8.4 Cost Management

- **Always-on service:** the connector runs continuously, so ORC_CDC_POOL (CPU_X64_M, 1 node) is billed for as long as the service runs. Suspend the service when replication isn't needed; the pool then suspends 60 s after it goes idle.
- **Warehouse:** ORC_CDC_WH (Gen2, SMALL) auto-suspends after 60 s without queries. It only suspends when no changes arrive for that long.
- **Right-sizing the pool:** the containers request 3.2 vCPU in total, so CPU_X64_S (3 vCPU) is too small unless the CPU requests in the service spec are lowered first.
- **Right-sizing the warehouse:** compare MERGE_MS and QUEUE_WAIT_MS in V_CDC_TABLE_STATS_MINUTE before changing the size.
- **Batching:** larger batches (APPLY_MAX_EVENTS) spread the fixed cost of each transaction across more rows.
- **Monitoring:** review credits for ORC_CDC_POOL and ORC_CDC_WH in the SNOWFLAKE.ACCOUNT_USAGE views.

8.5 Backup and Recovery

- **Time Travel and Fail-safe:** protect the replicated tables in ORC_INGEST according to the account's retention settings. Deletes are soft deletes, so deleted rows remain queryable (IS_DELETED = TRUE).
- **Restart recovery:** after a restart, suspension or crash, the service resumes from the positions in LOGS_ORC_INGEST.CDC_APPLY_STATE. Each row's _CDC_SEQ makes reprocessing a change harmless. This only works while Oracle still has the archive logs back to that position, so keep archive logs long enough to cover the longest expected outage.

- **No full reload:** if the needed logs are gone, the connector can't rebuild a table by itself. Re-sync it from Oracle with a separate one-off load, then resume CDC from a new start position (CDC_CONFIG.OLR_START).
- **Configuration:** export the CONFIG_ORC_INGEST tables, especially TABLE_FILTER_CONFIG, periodically as part of change control.

9 Testing and Validation

9.1 Test Approach

Testing is done in a non-production Snowflake account against a non-production Oracle database before the application is enabled in production. The Oracle database must have ARCHIVELOG mode and supplemental logging enabled. The same tests are repeated after each application upgrade.

Test results are checked in:

- the replicated tables (ORC_INGEST.<TABLE>);
- the log tables and views in LOGS_ORC_INGEST;
- the container logs (CORE.GET_SERVICE_LOGS).

9.2 Test Scenarios

ID	Scenario	Steps	Expected Result
TC-01	Connectivity and start-up	Bind the 3 references (Oracle secret, PEM secret, EAI), then call `CORE.CREATE_SERVICE()`	References show as bound. `GET_SERVICE_STATUS()` shows all 3 containers `READY`. `CDC_SERVICE_LOG` has a `STARTUP` event. Archive logs appear in `FSTREAM_AUDIT` as `Success`.
TC-02	Table creation	Register a new table in `TABLE_FILTER_CONFIG` and restart the service	`ORC_INGEST.<TABLE>` is created with the configured columns and the pipeline columns (`IS_DELETED`, `DELETED_AT`, `_CDC_SEQ`, `SNOWFLAKE_INGESTED_AT`, `SNOWFLAKE_MODIFIED_AT`).
TC-03	Insert	Insert rows in Oracle and commit	Rows appear in the target table within the agreed lag. `SNOWFLAKE_INGESTED_AT` and `SNOWFLAKE_MODIFIED_AT` are set. `CDC_BATCH_LOG` shows `STATUS = 'OK'`.
TC-04	Update	Update some rows (all columns, and only some columns)	The target reflects the new values. Columns that weren't changed keep their values. `SNOWFLAKE_MODIFIED_AT` advances; `SNOWFLAKE_INGESTED_AT` does not change.
TC-05	Delete	Delete rows in Oracle	The row stays, with `IS_DELETED = TRUE` and `DELETED_AT` set to the Oracle commit time.
TC-06	Re-insert a deleted key	Insert a row with a key that was deleted earlier	The same row comes back with `IS_DELETED = FALSE` and `DELETED_AT = NULL`. No duplicate row.
TC-07	Primary key update	Update the primary key value of a row	The old key is soft-deleted and the new key is created with the row's data.
TC-08	Transaction consistency	Commit one Oracle transaction that changes several rows in several tables. Also roll back a transaction.	All of the committed changes appear. Nothing from the rolled-back transaction appears.

ID	Scenario	Steps	Expected Result
TC-09	High volume	Run a bulk load in Oracle (e.g. 1M rows, several archive log switches)	All rows arrive. Batches stay within `APPLY_MAX_EVENTS`, and there are no out-of-memory restarts. `V_ARCHIVE_FILE_LATENCY` shows stable latency once the backlog clears.
TC-10	Data type coverage	Replicate a table that contains every mapped type	Values are converted according to Section 5.6. Any unmapped type becomes VARCHAR, with a warning in the log.
TC-11	NULLs and special values	Insert NULLs, empty strings, very long strings/CLOBs, extreme dates and timestamps, maximum-precision numbers	Values are preserved. A row that can't be converted goes to `CDC_DEAD_LETTER`; the rest of the batch commits.
TC-12	DDL: add / modify / rename column	Run `ALTER TABLE ... ADD`, `MODIFY` (widen), and `RENAME COLUMN` in Oracle	The change is applied to the target table. `CDC_AUDIT_DDL` shows `APPLIED`. Later changes fill the new or renamed column.

9.3 Acceptance Criteria

- All test scenarios pass in the target environment.
- **Data matches.** For every configured table, COUNT(*) WHERE NOT IS_DELETED in Snowflake matches COUNT(*) in Oracle when compared at a quiet point. Spot-checked key values match too.
- **No duplicates.** Each primary key appears once in each target table.
- **Latency.** Replication lag (LAST_LAG_MS in V_CDC_TABLE_SUMMARY) and TOTAL_LATENCY_MS in V_ARCHIVE_FILE_LATENCY stay within the agreed target under normal and peak load.
- **No unexplained errors.** No dead-letter rows or FATAL events are left without an explanation after the tests.
- **No visible credentials.** No passwords or keys appear in logs, configuration tables, container logs or application files.
- **Alerts work.** The consumer-configured alerts (Snowflake Alerts on CDC_SERVICE_LOG, CDC_BATCH_LOG, CDC_DEAD_LETTER, and lag) fire for a simulated failure. The connector only writes the log rows; it doesn't send notifications itself.
- **Cost.** Compute pool and warehouse credits are within the agreed limits.

9.4 Go-Live Checklist

#	Item	Status
1	Oracle readiness completed (Section 6.5): ARCHIVELOG mode, supplemental logging on every captured table, OLR database user and grants, SSH access to the archive log folder	
2	Archive log retention on Oracle covers the longest expected outage	
3	All 3 references bound (Oracle credentials, SFTP PEM key, EAI)	
4	All tables registered in TABLE_FILTER_CONFIG with the correct key, columns and types	

#	Item	Status
5	Start position agreed (OLR_START / START_SCN), and existing data loaded separately and reconciled	
6	CDC validated for every table: insert, update, delete, and DDL	
7	Service running continuously; alerts on lag, failed batches, dead-letter rows and FATAL events configured	
8	Role mapping (APP_ROLE grants) reviewed by security	
9	Runbook (Section 8) shared with the support team	

10 Troubleshooting Scenarios & Recommendations

This section outlines common technical issues that may arise during system operation along with recommended steps for diagnosis and resolution.

Scenario	Description	Possible Causes	Recommended Actions
Service not running	SPCS service stays in PENDING / SUSPENDED state, or a container is not READY	References not bound; compute pool ORC_CDC_POOL suspended or out of nodes; image pull failure	Bind all three references; run core.get_service_status() and core.get_service_logs(); resume with core.resume_service(), or run core.create_service() if the service was never created
Connectivity error	OpenLogReplicator cannot connect to Oracle, or SFTP cannot reach the host	Reference binding for a secret or the EAI missing or invalid; EAI created without ALLOWED_AUTHENTICATION_SECRETS; expired Oracle password; wrong host, port or service name	Re-bind the application references in Settings; validate the secret values; confirm the network rule has host:1521 and host:22; check olr_config.ORACLE_SERVER; restart the service
Archive logs not copied	ec2-ingest-container copies no files, or fstream_error_log has rows	Wrong SFTP_HOST, SFTP_USER or REMOTE_FOLDER in fstream_config; invalid PEM key; SSH user cannot read the archive folder	Review fstream_error_log and the container logs; correct fstream_config or the sftp_pem_key secret; restart the service
Archive logs missing	OpenLogReplicator cannot continue after an outage or a new start	Archive logs back to the saved position or START_SCN were deleted on the Oracle host	Keep archive logs longer than the longest expected outage; if they are gone, restart from a later position and load the gap separately
Ingestion (batch) failure	cdc_batch_log has ERR rows, or a table stops advancing	Data type mismatch with COLUMN_TYPES; value too long; warehouse or network errors	Review ERROR_MESSAGE in cdc_batch_log and rows in cdc_dead_letter; fix the type mapping in table_filter_config; restart. Network and warehouse errors are retried automatically
DDL not applied	A schema change in Oracle does not show up in Snowflake	RENAME TABLE; a MODIFY COLUMN that cannot be widened in place; a skipped DROP	Check cdc_audit_ddl for ACTION_STATUS = 'FAILED'; apply the change by hand; update table_filter_config; restart
Table not replicated	An Oracle table has no target table, or receives no changes	Missing row or IS_ACTIVE = FALSE in table_filter_config; name not in capital letters; wrong ORACLE_OWNER; supplemental logging not enabled	Verify the table_filter_config entry; enable supplemental logging in Oracle; restart; run core.grant_table_access()

Scenario	Description	Possible Causes	Recommended Actions
Slow apply / growing lag	LAG_MS or QUEUE_WAIT_MS keeps growing	Warehouse too small; too few apply workers; very large tables	Size up ORC_CDC_WH, or add clusters or raise MAX_CONCURRENCY_LEVEL; use v_cdc_table_stats_minute and v_archive_file_latency to find the slow step; consider clustering by primary key or Search Optimization
Duplicate rows	More than one row for a primary key	APPLY_MODE = 'INSERT_MERGE' with keys reused after a delete, or an initial load overlapping the stream	Set APPLY_MODE back to 'MERGE'; restart; remove the duplicates

10.1 Additional Error Reference

Error	Cause	Resolution
ORA-12170 / connection timed out	Network rule, firewall, or security group blocks access to the Oracle listener or SSH	Confirm host:1521 and host:22 in the network rule and allow Snowflake egress on the Oracle side
ORA-12514 listener does not know of service	Wrong service name in ORACLE_SERVER	Correct olr_config.ORACLE_SERVER (//host:port/service) and restart the service
ORA-01017 invalid username/password	Wrong or rotated password; wrong user	Update the oracle_credentials secret and restart the service
ORA-28000 account is locked	OLR database user locked after failed logins or by policy	Unlock the user in Oracle, then restart the service
ORA-01031 insufficient privileges	OLR user lacks the grants it needs to read the dictionary and logs	Grant the required privileges to the OLR user
SSH authentication failed	Wrong SFTP_USER, or invalid PEM key (missing BEGIN/END lines)	Fix fstream_config.SFTP_USER or the sftp_pem_key secret; restart
SFTP permission denied / no such file	SSH user cannot read REMOTE_FOLDER, or wrong path	Grant read access on the archive folder or correct REMOTE_FOLDER
Archive log not found for the start or saved position	Archive logs deleted on the Oracle host before they were copied	Keep archive logs longer; if they are gone, restart from a later position and load the gap separately
Supplemental logging missing / columns empty	Supplemental logging not enabled on the table	Enable at least primary key, ideally (ALL) COLUMNS, supplemental logging
Compute pool suspended / service PENDING	ORC_CDC_POOL suspended or node limit reached	Resume the pool and service; review node limits; check core.get_service_status()

Error	Cause	Resolution
Numeric value out of range	Oracle NUMBER precision exceeds the type in COLUMN_TYPES	Widen the type in table_filter_config or map it to VARCHAR / FLOAT; rejected rows go to cdc_dead_letter
String is too long	Value longer than the target VARCHAR(n)	Widen the column type in table_filter_config; check cdc_dead_letter
DDL FAILED in cdc_audit_ddl	RENAME TABLE, or MODIFY COLUMN that cannot be widened in place	Apply the change by hand, update table_filter_config, restart
Duplicate rows for a primary key	APPLY_MODE = 'INSERT_MERGE' with reused keys or an overlapping initial load	Switch back to MERGE, restart, and remove the duplicates

10.2 Escalation Path

Level	Responsible	Action
L1	Consumer Data Engineer	Review AUDIT_LOG and ERROR_LOG, check references, service and compute pool status
L2	Consumer Admin / Oracle DBA	Resolve privilege, network, credential, and source-side issues
L3	Pi-Snow Support Team	Investigate application defects; provide a fix or patch

11 Glossary

This section defines key terms, acronyms, and technical language used throughout the document. It ensures a common understanding among all contributors and stakeholders.

Term / Acronym	Definition
Change Data Capture (CDC)	A process of capturing INSERT, UPDATE, and DELETE operations from an Oracle database and replicating those changes to Snowflake.
OpenLogReplicator (OLR)	An open-source tool that reads Oracle redo and archived redo logs, extracts database changes, and publishes them as CDC events.
Redo Logs	Oracle database logs that record changes made to the database and serve as the primary source for CDC event extraction.
Archived Redo Logs	Copies of filled redo logs retained by Oracle for recovery and used by the connector to process database changes.
Snowpark Container Services (SPCS)	Snowflake's managed platform for deploying and running containerized applications and services.
Snowflake Native App	A Snowflake application packaging and distribution framework that enables applications to be installed and operated in consumer accounts.
SFTP (Secure File Transfer Protocol)	A secure protocol used to transfer archived redo log files from the Oracle environment to Snowflake.
Reference Binding	A Native App mechanism that allows an application to access consumer-approved resources, such as secrets and external access integrations.
External Access Integration (EAI)	A Snowflake object that controls outbound network access from an application to approved external endpoints.
Supplemental Logging	An Oracle database feature that adds required column information to redo records to help identify affected rows during CDC processing.
System Change Number (SCN)	An Oracle-generated logical sequence number that identifies a point in the database change history and is used to track CDC progress.
Checkpoint	A persisted processing position that allows the connector to resume CDC processing from the last successfully completed point after an interruption.
Audit Logging	Recording CDC processing activity, batches, outcomes, and errors to support monitoring, reconciliation, and troubleshooting.
Transaction-Aware Processing	Processing CDC events while preserving the original Oracle transaction boundaries, ensuring that changes belonging to a transaction are handled together.
Idempotency	A property that ensures reprocessing the same CDC event does not cause duplicate or incorrect changes in the target. The _CDC_SEQ value helps prevent older changes from overwriting newer ones.

12 Reference Artifacts & Editable Links

This section provides direct access to project artifacts and reference documentation. These resources are maintained in shared locations to ensure version control and collaborative updates.

Asset type	Description	Official doc link
Snowflake Documentation	Official Snowflake documentation portal	https://docs.snowflake.com
Native Apps Framework	Building and distributing Snowflake Native Apps	https://docs.snowflake.com/en/developer-guide/native-apps/native-apps-about
Native App References	Requesting consumer objects (secrets, EAI) through references	https://docs.snowflake.com/en/developer-guide/native-apps/requesting-refs
Snowpark Container Services (SPCS)	Overview and concepts for running containerized workloads	https://docs.snowflake.com/en/developer-guide/snowpark-container-services/overview
Service Specification	Reference for the multi-container service spec (containers, volumes, resources)	https://docs.snowflake.com/en/developer-guide/snowpark-container-services/specification-reference
Compute Pools	Configuring and managing compute pools	https://docs.snowflake.com/en/developer-guide/snowpark-container-services/working-with-compute-pool
Image Repository	Storing and managing container images in Snowflake	https://docs.snowflake.com/en/developer-guide/snowpark-container-services/working-with-registry-repository
Warehouses	Creating and configuring the warehouse used for MERGE operations	https://docs.snowflake.com/en/sql-reference/sql/create-warehouse
Hybrid Tables	Row-level locking table type used for CDC_APPLY_STATE	https://docs.snowflake.com/en/user-guide/tables-hybrid
External Access Integration	Secure outbound network access from Snowflake	https://docs.snowflake.com/en/sql-reference/sql/create-external-access-integration
Network Rules	Defining egress network rules (Oracle 1521, SSH 22)	https://docs.snowflake.com/en/sql-reference/sql/create-network-rule
Snowflake Secrets	Secure storage and usage of credentials	https://docs.snowflake.com/en/sql-reference/sql/create-secret
Stored Procedures	Procedures used for service lifecycle and credential retrieval	https://docs.snowflake.com/en/sql-reference/sql/create-procedure
Snowflake Connector for Python	Python driver used by the table writer and ec2-ingest containers	https://docs.snowflake.com/en/developer-guide/python-connector/python-connector

Asset type	Description	Official doc link
Oracle Database Documentation	Official Oracle manuals (archive log mode, supplemental logging, LogMiner/redo concepts)	https://docs.oracle.com/en/database/
OpenLogReplicator	Open-source Oracle redo log reader used by `olr-container`	https://github.com/bersler/OpenLogReplicator
Paramiko	Python SSH/SFTP library used by `ec2-ingest-container`	https://docs.paramiko.org/
Protocol Buffers	Serialization format used between OpenLogReplicator and the table writer	https://protobuf.dev/

13 Appendix

This section includes supporting information, usage guidelines, and references.

13.1 Usage Guidelines for Contributors

- Follow established naming conventions for Snowflake objects, application roles, and manifest-defined references.
- Ensure all ingestion and transformation jobs write detailed execution telemetry to the internal LOGS_PG_INGEST schema.
- All container images must undergo a mandatory security scan and patch update before a new Application Version or Patch is published.
- Do not modify the setup.sql or manifest.yml files without an approved change request and versioning plan.

13.2 Environment Deployment Guide

This section provides guidance for installing and configuring the Pi-Ingest Oracle application instance within a consumer environment.

- **Snowflake account**
  - Snowpark Container Services (SPCS) and hybrid tables available in the target region.
  - An edition that supports Native Apps with containers.
- **Administrative privileges**
  - A role (for example ACCOUNTADMIN) that can create secrets, network rules and external access integrations.
  - That role can also approve the app's CREATE COMPUTE POOL and CREATE WAREHOUSE privileges.
- **Security objects** (in the consumer account)
  - An Oracle credential secret (TYPE = PASSWORD).
  - An SSH private key secret (TYPE = GENERIC_STRING, holding the full PEM file).
- **Network governance**
  - A network rule allowing egress to <oracle_host>:1521 and <sftp_host>:22.
  - An external access integration with ALLOWED_AUTHENTICATION_SECRETS listing both secrets.
- **Oracle source**
  - ARCHIVELOG mode enabled.
  - Supplemental logging on every captured table: at least the primary key; (ALL) COLUMNS is recommended.
  - A database user for OpenLogReplicator with the grants it needs.
  - Archive log retention long enough to cover the longest expected outage.
- **Archive log host**
  - SSH/SFTP access with a PEM key to the Oracle archive log folder.

13.2.2 Deployment Steps

The deployment of a Snowflake Native App follows a provider-consumer model. The following steps outline the end-to-end process:

- **Image provisioning (provider).** Build and push the three images (`olr`, `table_writer`, `ec2_ingest`) to the image repository `/orc_ingest/landing_orc_ingest/orc_img/` in the application package.
- **Version creation (provider).** Package `setup.sql`, `manifest.yml`, `service_spec.yaml` and `README.md`, then create a new application version or patch. Share it through a Marketplace or private listing.
- **App installation (consumer).** Install the app from the listing and approve the requested privileges (`CREATE COMPUTE POOL`, `CREATE WAREHOUSE`). The setup script then creates the schemas, configuration and log tables, the stage, `ORC_CDC_POOL` (CPU_X64_M) and `ORC_CDC_WH` (Gen2, SMALL).
- **Reference binding (consumer).** Bind the three required references in Snowsight or with SQL:
  1. Oracle Database Credentials → the Oracle secret
  2. SFTP PEM Key Secret → the PEM secret
  3. External Access Integration → the EAI
- No warehouse or compute pool is bound, because the app creates and owns both.
- **Configuration (consumer).** Set `ORACLE_SERVER` in `OLR_CONFIG`, set the SFTP host, user and folder in `FSTREAM_CONFIG`, and register the tables in `TABLE_FILTER_CONFIG` (see Section 7).
- **Service start (consumer).** The service starts once the references are bound. After configuration changes, call `core.suspend_service()` and then `core.resume_service()`. If the service doesn't exist yet, call `core.create_service()`.

13.2.3 Environment Configuration

One continuous service. A single SPCS service, CORE.ORACLE_CDC_SERVICE, runs three containers on ORC_CDC_POOL.

They run continuously; there are no schedules:

- ec2-ingest-container copies archive logs over SFTP onto the stage;
- olr-container runs OpenLogReplicator;
- qtp-container applies the changes to the target tables.

Metadata management. All settings are in CONFIG_ORC_INGEST:

- CDC_JOB_REGISTRY: jobs.
- CDC_CONFIG: writer settings (queue and batch limits, APPLY_MODE, start position).
- OLR_CONFIG: OpenLogReplicator settings.
- FSTREAM_CONFIG: SFTP transfer settings.
- TABLE_FILTER_CONFIG: the tables to replicate.

Role mapping. The app has one application role, APP_ROLE. It has read/write access to the configuration tables, read access to the log tables and views, and can run the service procedures. Grant it to the consumer's roles with GRANT APPLICATION ROLE <app_name>.app_role TO ROLE <role>, and run core.grant_table_access() whenever new target tables appear.

13.2.4 Validation

After installation, check:

- **References:** all three (Oracle secret, PEM secret, EAI) are bound.
- **Service status:** CALL core.get_service_status() shows olr-container, qtp-container and ec2-ingest-container as READY.
- **Archive log transfer:** LOGS_ORC_INGEST.FSTREAM_AUDIT has Success rows, and FSTREAM_ERROR_LOG has no new errors.
- **Replication health:**

1. LOGS_ORC_INGEST.V_CDC_TABLE_SUMMARY shows recent LAST_BATCH_AT and an acceptable LAST_LAG_MS.
2. CDC_BATCH_LOG has no repeated ERR rows.
3. The LWM row in CDC_APPLY_STATE keeps advancing.

- **End-to-end test:** insert, update and delete a row in Oracle. Within the expected lag, the change appears in ORC_INGEST.<TABLE>; the deleted row shows IS_DELETED = TRUE.
- **Logs:** CDC_SERVICE_LOG has a STARTUP event, and the container logs are available through core.get_service_logs(0, '<container>').
- **Access:** roles granted APP_ROLE can query the target and log tables; other roles are denied.

13.3 Frequently Asked Questions

Question	Answer
Does π-Ingest need software on the Oracle server?	**No software is installed. The source needs ARCHIVELOG mode, supplemental logging, a database user for OpenLogReplicator, and SSH/SFTP read access to the archive log folder.**
Can the provider see our credentials or data?	**No. Secrets belong to the consumer and are bound by reference. Data stays in the consumer's Snowflake account.**
Does it support change data capture?	**Yes. It's log-based CDC: OpenLogReplicator reads the Oracle redo and archive logs, and inserts, updates, deletes and supported DDL are applied continuously.**
Does it do an initial (full) load?	**No. It captures changes from the configured start position (OLR_START / START_SCN). Rows that existed before that point must be loaded separately.**
How are deleted rows handled?	**They're soft deletes: the row stays with IS_DELETED = TRUE and DELETED_AT set to the Oracle commit time. Read live rows with WHERE NOT IS_DELETED.**
How are schema changes handled?	**ADD COLUMN, MODIFY (widening), RENAME COLUMN and TRUNCATE are applied. DROP TABLE and DROP COLUMN are logged but not applied, so data is kept. RENAME TABLE needs manual action. Everything is recorded in CDC_AUDIT_DDL.**
Are transactions preserved?	**Yes. Changes are applied per table along transaction boundaries. Each table's position is committed in the same transaction as its data.**
What happens after a restart or outage?	**The service resumes from the positions saved in CDC_APPLY_STATE, with no loss or duplicates. Oracle must still have the archive logs back to that position.**
How do I add a new table?	**Enable supplemental logging on it, add a row to TABLE_FILTER_CONFIG (table, owner, key, columns, types), and restart the service. No code change is needed.**
How is batch size controlled?	**Through configuration in CDC_CONFIG: APPLY_MAX_EVENTS caps each table batch, and MAX_QUEUE_COUNT / MAX_QUEUE_BYTES / TABLE_INBOX_MAX limit buffering.**
How do I stop ingestion immediately?	**CALL core.suspend_service(). As an account-level kill switch, you can also disable the external access integration.**
What are the metadata columns on target tables?	**IS_DELETED, DELETED_AT, _CDC_SEQ (Oracle position of the last applied change), SNOWFLAKE_INGESTED_AT and SNOWFLAKE_MODIFIED_AT (UTC).**

Question	Answer
Which Oracle versions are supported?	**The Oracle versions supported by the bundled OpenLogReplicator release (OLR_CONFIG.OLR_VERSION). Confirm the exact list with the OpenLogReplicator documentation for that release before publishing.**
Are RAC or Data Guard standby sources supported?	**Not verified in this version. Confirm before publishing.**

13.4 Roadmap Considerations

- Initial (snapshot) load integrated with CDC, so new tables don't need a separate load.
- Built-in alerting on lag, failed batches, dead-letter rows and FATAL events.
- Automatic handling of RENAME TABLE, and a notification when DDL is skipped.
- Tools for reprocessing dead-letter rows.
- Built-in data-quality rules on replicated tables.
- A status dashboard for consumers, built on the latency and summary views.
- Additional source connectors using the same framework.

13.5 Revision History

Version	Date	Author	Description
01	Initial release	Pibythree Developers	First version of the Pi-Ingest Oracle CDC document

13.6 Document Review and Sign-off

Role	Name	Signature	Date
Author			
Technical Reviewer			
Security Reviewer			
Project Manager			

14 Support & Contact Information

14.1 Technical Support

For issues related to ingestion failures, connectivity troubleshooting, or bug reporting, the Pi-Snow Support Team is available to assist:

- **Email:** Admin@Pibythree.com, Alliance@Pibythree.com

Note: Because this is a Snowflake Native App, the support team can't see your account. Before contacting them, please check the following and include the relevant output in your request. This helps the team resolve it faster.

- **Service status:** `CALL <app_name>.core.get_service_status();`
- **Container logs** for the container that's affected (`olr-container`, `qtp-container` or `ec2-ingest-container`): `CALL <app_name>.core.get_service_logs(0, '<container_name>');`
- **Failed batches:** rows with `STATUS = 'ERR'` in `LOGS_ORC_INGEST.CDC_BATCH_LOG`
- **Dead-letter rows:** `LOGS_ORC_INGEST.CDC_DEAD_LETTER`
- **Service events:** `FATAL` and `RECONNECT` events in `LOGS_ORC_INGEST.CDC_SERVICE_LOG`
- **Schema changes not applied:** `LOGS_ORC_INGEST.CDC_AUDIT_DDL`
- **Archive log transfer errors:** `LOGS_ORC_INGEST.FSTREAM_ERROR_LOG`

Please also include:

- the app version;
- the affected table names;
- the approximate time the issue started, in UTC.

Never send passwords, PEM keys or secret values.

14.2 Business & Sales Inquiries

For inquiries regarding the broader Pi-Snow roadmap, licensing for additional Oracle objects, or custom integration needs:

- **Contact:** Alliance@pibythree.com
- **Website:** www.pibythree.com

14.3 Documentation & Feedback

To access the latest version of this document or to provide feedback on the application's performance, visit the Pi Accelerators by PibyThree documentation portal.

14.4 Customization & Full Product

This can be configured and customized to support a wide range of structured databases / RDBMS. For implementation, please contact us.