# IMG_256

# Airbyte and PostgreSQL Data Ingestion Documentation

**Table of Contents**

1. [Project Overview](#908pk4lyrlah)
2. [System Architecture](#1a28fgal9fyb)
3. [PostgreSQL Database Configuration](#ftrbgjg2g25q)
4. [Airbyte Configuration and Setup](#18ge6dns1os1)
5. [Data Flow Logic](#codkv9hpnra0)
6. [Operational Guidance](#pqdq5t89sonk)
7. [Testing and Validation](#g2ukvc2owq0j)
8. [Troubleshooting Guide](#tz98r3m1wvwk)
9. [Performance Optimization](#52r1wqu928oz)
10. [Security and Best Practices](#f1zwqzsj6nyo)

## Project Overview

This PoC project was implemented in the span of 2 Weeks (25-07-2025 to 08-08-2025) by **Yatin Kumar Reddy K** & **Sriman Satwik Reddy Chinnam**, both being B.Tech students from Mahindra University, Hyderabad. This documentation details the design, configuration, and operation of a resilient data pipeline leveraging **Airbyte** for scheduled, incremental data ingestion from a PostgreSQL source database. Raw and transaction records are synchronized using Change Data Capture (CDC), written to staging tables and immutable raw storage, and made available for downstream processing (e.g., Apache NiFi) with high reliability and replay guarantees.

**Key Features**

* **CDC-Based Ingestion**: Uses PostgreSQL Write-Ahead Log for real-time change detection.
* **Dual Storage Strategy**: Raw data written to both staging schema and immutable object storage.
* **Webhook Integration**: Automatic triggering of downstream processing workflows.
* **Replay Capability**: Immutable micro-batches enable full pipeline replay and audit trails.

## System Architecture

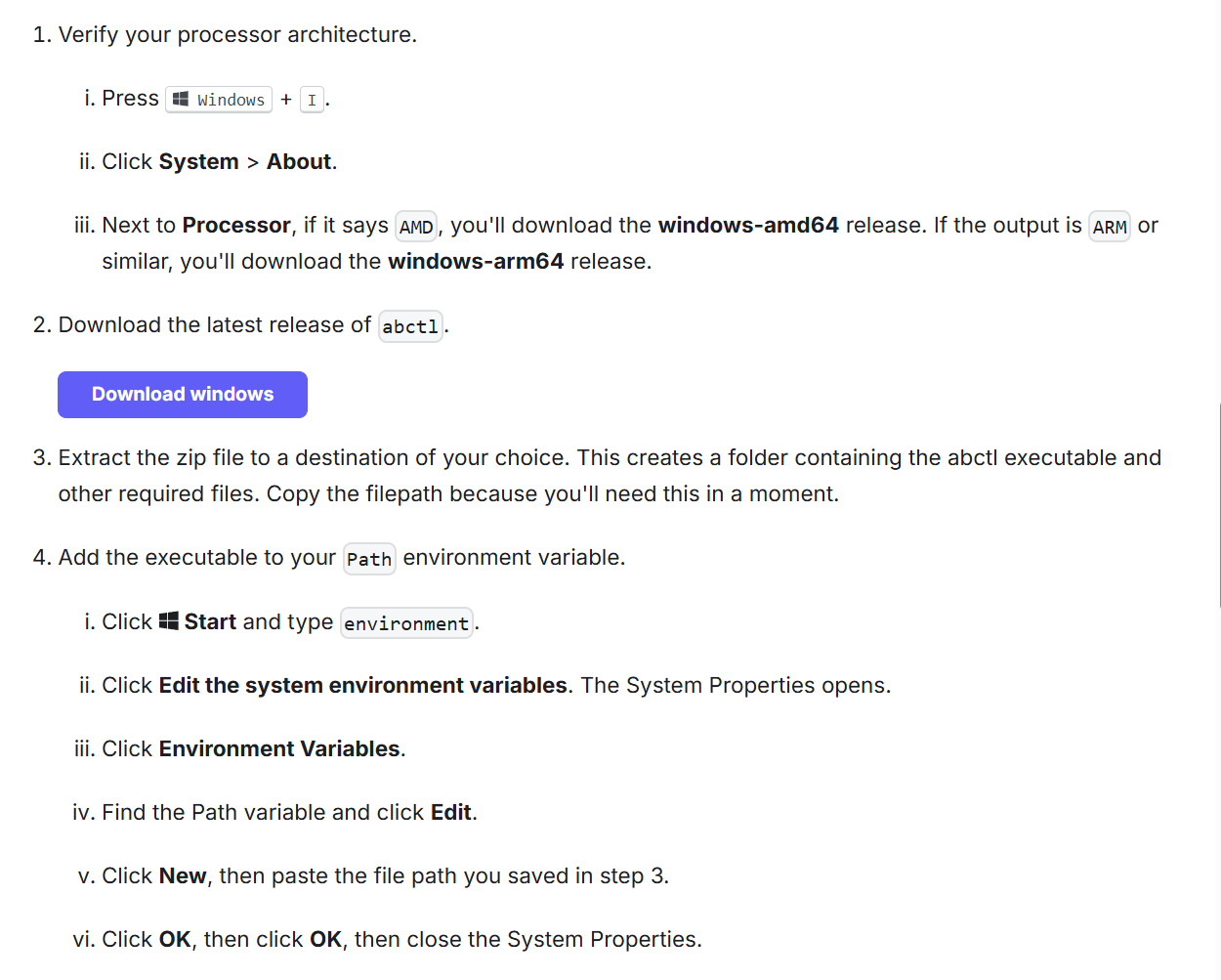
The PoC establishes a resilient two-step pipeline built on Airbyte and Apache NiFi. Airbyte pulls both master and transaction streams on a daily schedule, writing each record to a PostgreSQL staging\_ods schema and to an immutable object-storage folder tree (/raw). When a run finishes it emits a webhook that triggers NiFi.

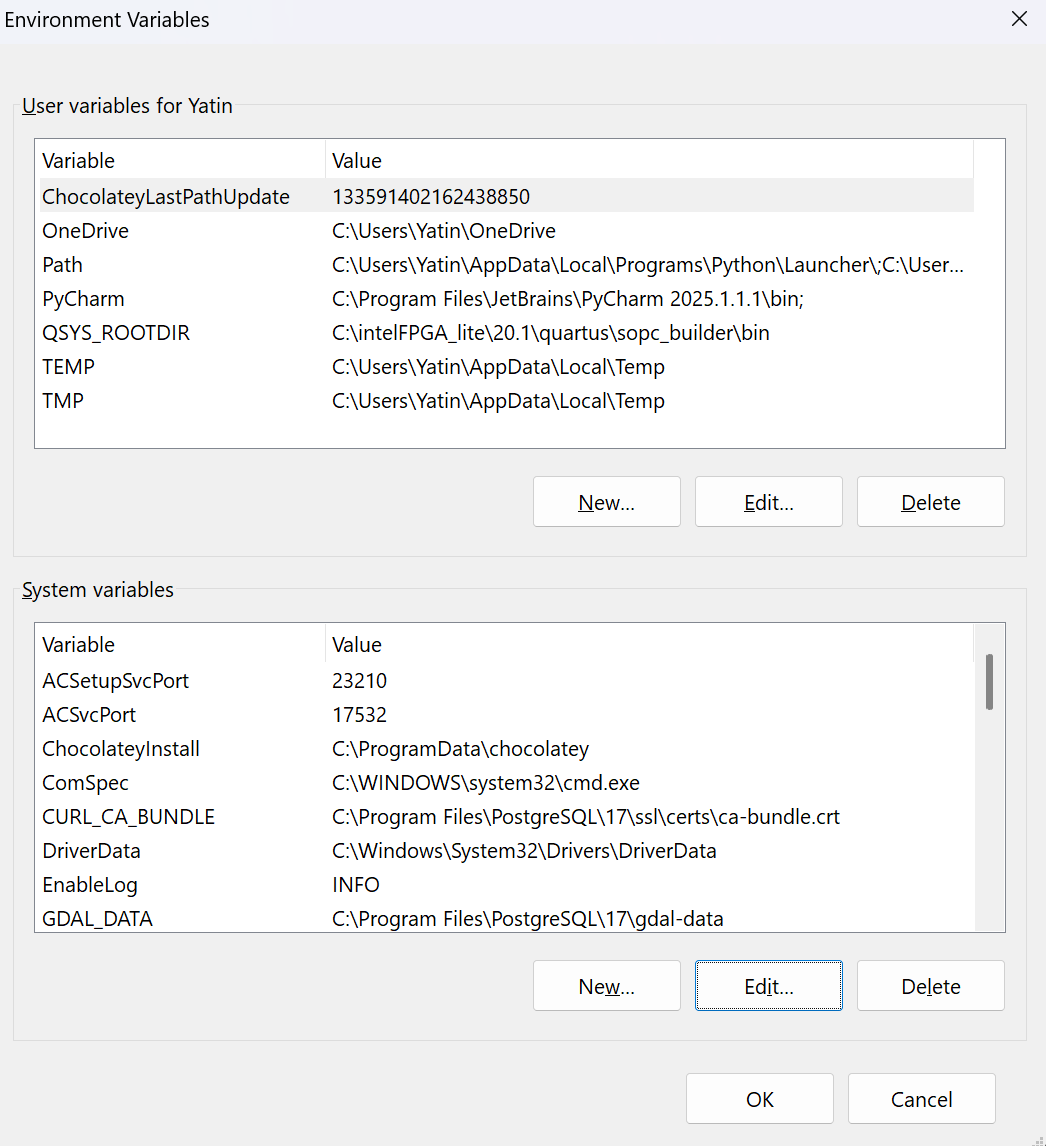
* **PostgreSQL source database** receives OLTP or transactional updates.
* **Airbyte** connects as an external agent, using CDC (Write-Ahead Log) to detect changes.
* Data batches are written to:
  + **staging\_ods schema** (PostgreSQL table for raw ingestion)
  + **raw object storage** using partitioned paths (/raw/<stream>/<YYYY-MM-DD>/<HH>/)
* **Webhook triggers** notify NiFi when ingestion completes.

**Airbyte CLI Installation and Environment Setup (Windows)**

To run Airbyte locally using the abctl CLI tool, install and configure it as follows:

1. **Verify your processor architecture:**
   * Press Windows + I.
   * Click **System** > **About**.
   * Next to **Processor**, if it says *AMD*, download the **windows-amd64** release. If it says *ARM* or similar, download the **windows-arm64** release.
2. **Download the latest release of abctl**  
   *(see Airbyte’s official*[*OSS Quickstart*](https://docs.airbyte.com/platform/using-airbyte/getting-started/oss-quickstart)*for download link)*
3. **Extract the ZIP file to your preferred directory**  
   This creates a folder with the abctl executable and other files. **Copy the folder path** for use in the next step.
4. **Add abctl to your system PATH:**
   * Click **Start** and type environment.
   * Click **Edit the system environment variables**.
   * Click **Environment Variables**.
   * Under “User variables”, select the **Path** variable and click **Edit**.
   * Click **New**, then paste the file path where you extracted abctl.
   * Click **OK** on all dialog boxes to save and close.





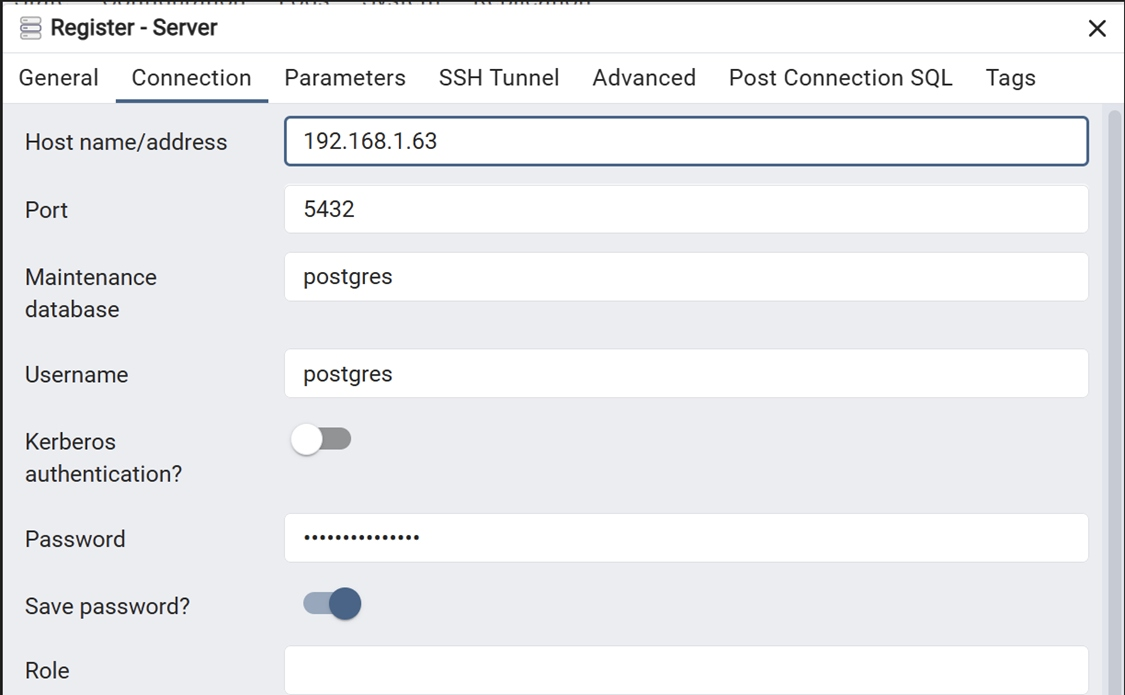
Full step-by-step Airbyte self-managed setup:  
<https://docs.airbyte.com/platform/using-airbyte/getting-started/oss-quickstart>

## PostgreSQL Database Configuration

### PostgreSQL Server Connection Settings

**Server Registration and Connection**

* **Host name/address**: 192.168.1.63 (**IPV4 ADDRESS**)
* **Port**: 5432
* **Maintenance database**: postgres
* **Username**: postgres
* **Authentication**: Password-based
* **Kerberos**: Not used
* **Save password**: Enabled



**NOTE:**  
The IPv4 address of your Windows machine used in your PostgreSQL and Airbyte source connection configuration may change each time the device is restarted or connected to a different network (for example: Wi-Fi reconnections or after reboot).

Before starting Airbyte or PostgreSQL, always check your current IPv4 address:

1. Go to Settings > Network & internet > Wi-Fi > [Your Active Network].
2. Look for the “IPv4 address” field (see figure below).

Figure: Finding your current IPv4 address in Windows Wi-Fi network properties (e.g., 192.168.0.104).

Be sure to update the “Host” field in both the Airbyte and PostgreSQL connector settings to match this address.  
This is required for the services to connect if the address changes after device/network events.

Before starting Airbyte or PostgreSQL:

* Always check your current IPv4 address by opening Command Prompt and running:

**“ipconfig”**

* Look for the IPv4 Address field in the output (example: 192.168.0.104).
* Alternatively, you can check under Windows Settings > Network & Internet > Wi-Fi > [Your Network] where the IPv4 address is listed.

Make sure to update the **Host** field in both your PostgreSQL and Airbyte source connector configurations to reflect the current IP address to maintain connectivity.

### Database and Schema Layout

**Database Structure**

* **Database**: scms\_db
* **Schemas**:
  + airbyte\_internal – Airbyte system tables
  + public – Main application tables including:
    - scms\_data\_1 (target for UPSERT)
    - scms\_data\_2 (Airbyte output)
  + staging\_ods – Raw operational data store (for ingestion batches)

*Insert database schema structure view here*

### Table Creation and Indexing

**Source Table (Airbyte Output)**

*-- Create source table with Airbyte metadata columns*  
**CREATE** **TABLE** **public**.scms\_data\_2 (  
 **id** INTEGER **PRIMARY** **KEY**,  
 name VARCHAR(255),  
 email VARCHAR(255),  
 phone VARCHAR(20),  
 address TEXT,  
 created\_date DATE,  
 *-- Airbyte metadata columns*  
 \_airbyte\_raw\_id UUID **NOT** **NULL**,  
 \_airbyte\_extracted\_at TIMESTAMP **WITH** TIME ZONE **NOT** **NULL**,  
 \_airbyte\_generation\_id BIGINT,  
 \_airbyte\_meta JSONB  
);  
  
*-- Create index for efficient incremental processing*  
**CREATE** **INDEX** idx\_scms\_data\_2\_extracted\_at **ON** **public**.scms\_data\_2(\_airbyte\_extracted\_at);  
**CREATE** **INDEX** idx\_scms\_data\_2\_id **ON** **public**.scms\_data\_2(**id**);

**Target Table (NiFi Output)**

*-- Create target table for processed data*  
**CREATE** **TABLE** **public**.scms\_data\_1 (  
 **id** INTEGER **PRIMARY** **KEY**,  
 name VARCHAR(255),  
 email VARCHAR(255),  
 phone VARCHAR(20),  
 address TEXT,  
 created\_date DATE,  
 *-- Processing metadata*  
 updated\_at TIMESTAMP **WITH** TIME ZONE **DEFAULT** CURRENT\_TIMESTAMP  
);  
  
*-- Create index for efficient UPSERT operations*  
**CREATE** **INDEX** idx\_scms\_data\_1\_id **ON** **public**.scms\_data\_1(**id**);

### Database User and Permissions

**Create dedicated user for NiFi and Airbyte ETL**

*-- Create dedicated user for pipeline operations*  
**CREATE** USER nifi\_user **WITH** **PASSWORD** 'secure\_password';  
  
*-- Grant necessary permissions*  
**GRANT** **USAGE** **ON** **SCHEMA** **public** **TO** nifi\_user;  
**GRANT** **SELECT** **ON** **public**.scms\_data\_2 **TO** nifi\_user;  
**GRANT** **INSERT**, **UPDATE**, **DELETE** **ON** **public**.scms\_data\_1 **TO** nifi\_user;  
**GRANT** **USAGE**, **SELECT** **ON** **ALL** SEQUENCES **IN** **SCHEMA** **public** **TO** nifi\_user;

**Verify Database Setup**

*-- Check table structure*  
\d **public**.scms\_data\_2  
\d **public**.scms\_data\_1  
  
*-- Verify sample data with Airbyte columns*  
**SELECT** **id**, name, \_airbyte\_extracted\_at   
**FROM** **public**.scms\_data\_2   
**ORDER** **BY** \_airbyte\_extracted\_at **DESC**   
**LIMIT** 5;

## Airbyte Configuration and Setup

### Source Connector Configuration

**Basic Source Settings**

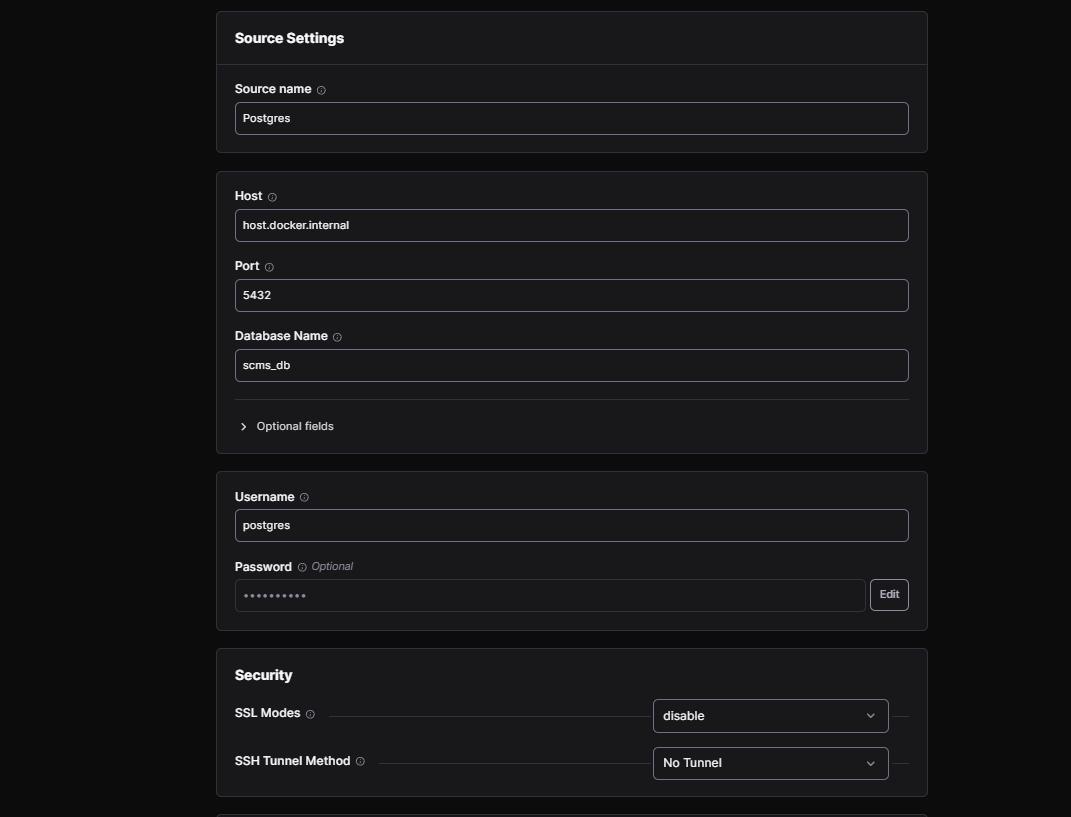
Airbyte connects to PostgreSQL using CDC with the following configuration:

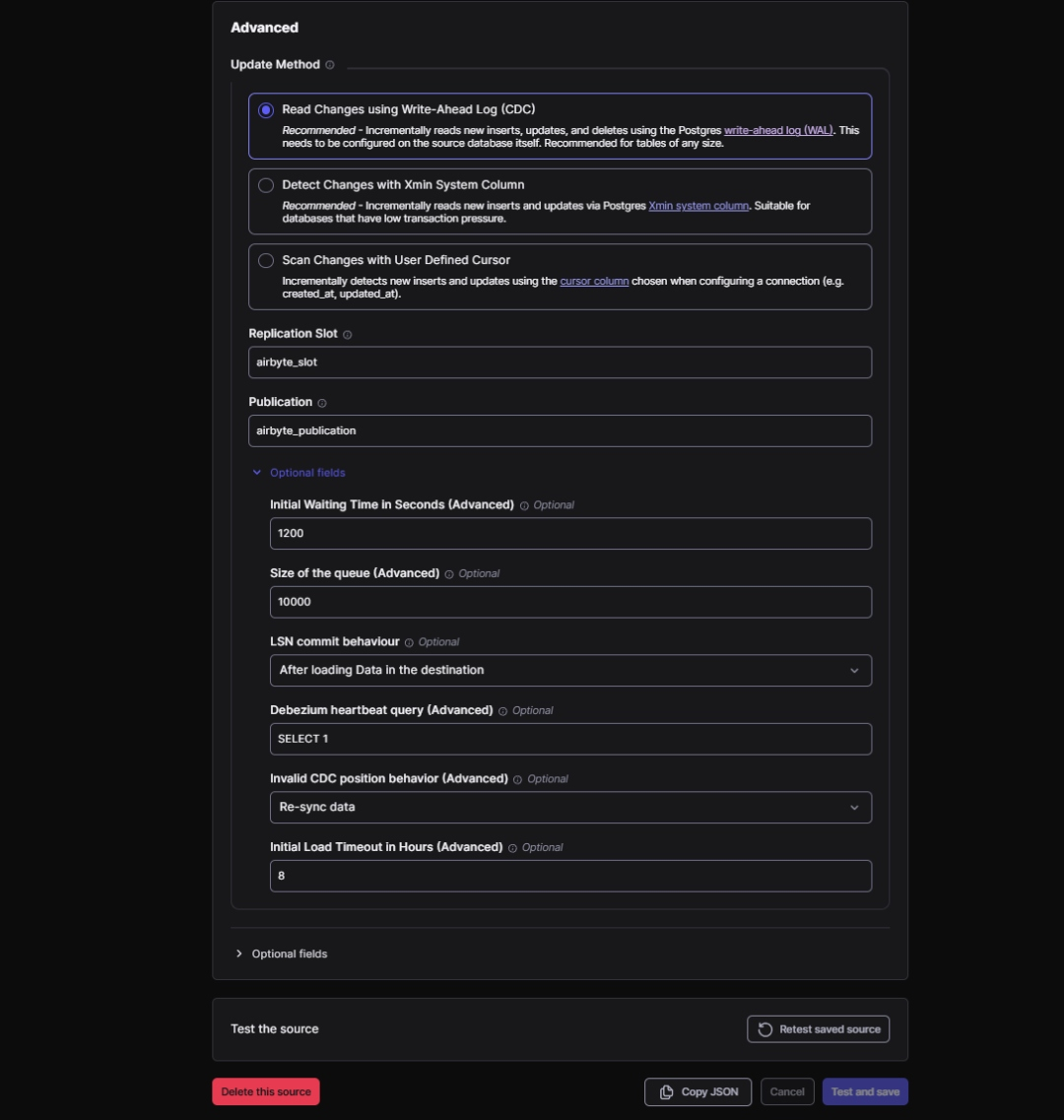
* **Source Name**: Postgres
* **Host**: 192.168.1.63 (**IPV4 ADDRESS**)
* **Port**: 5432
* **Database Name**: scms\_db
* **Username**: postgres
* **Password**: [configured securely]
* **SSL Mode**: Disabled
* **SSH Tunnel**: No Tunnel

### Advanced Source Options

**CDC Configuration with Write-Ahead Log**

* **Update Method**: Read Changes using Write-Ahead Log (CDC) - **Recommended**
* **Replication Slot**: airbyte\_slot
* **Publication**: airbyte\_publication
* **Initial Waiting Time**: 1200 seconds
* **Queue Size**: 10000
* **LSN Commit Behaviour**: After loading data in the destination
* **Debezium Heartbeat Query**: SELECT 1
* **Initial Load Timeout**: 8 hours

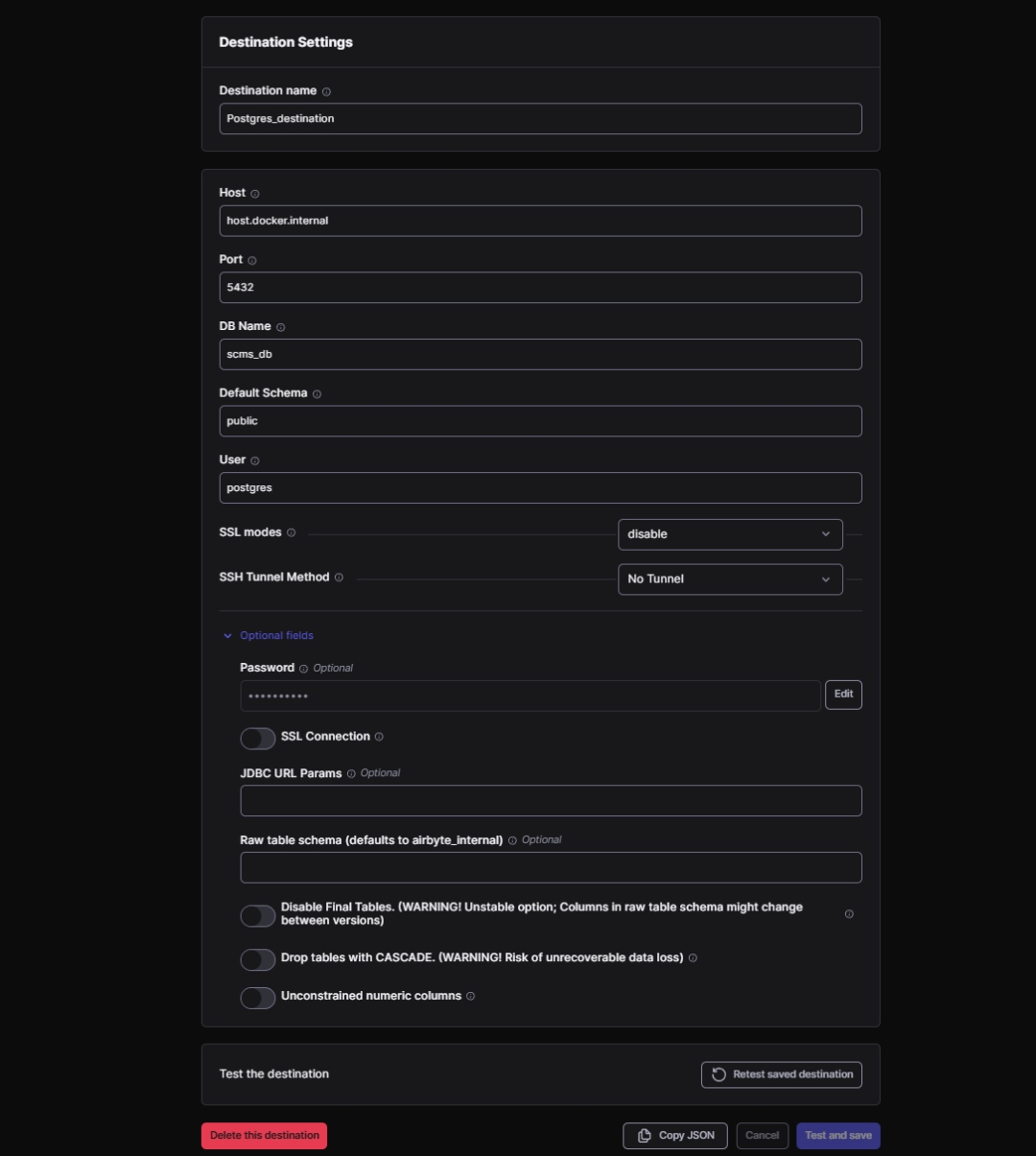




### Destination Configuration

**PostgreSQL Destination Settings**

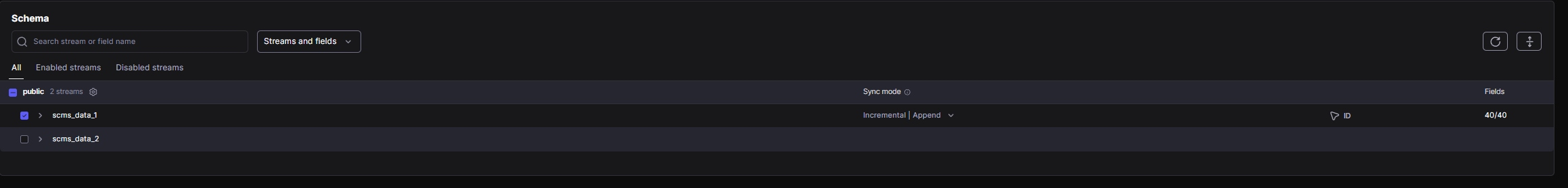
* **Destination Name**: Postgres\_destination
* **Host**: 192.168.1.63 (**IPV4 ADDRESS**)
* **Port**: 5432
* **DB Name**: scms\_db
* **Default Schema**: public
* **User**: postgres
* **SSL Mode**: Disabled
* **SSH Tunnel Method**: No Tunnel



### Schema & Sync Mode Configuration

**Stream Configuration**

* **Enabled Streams**:
  + scms\_data\_1 (target, Incremental/Append)
  + scms\_data\_2 (source)
* **Sync Mode**: Incremental | Append
* **Primary Key for Incremental Sync**: id
* **Cursor Field**: \_airbyte\_extracted\_at (for efficient batch tracking)



## 

## Data Flow Logic

### Pipeline Logic Overview

**1. Airbyte CDC Extraction** - Airbyte polls source PostgreSQL at scheduled intervals using CDC - All changed/new rows (as determined by WAL) are extracted and synchronized - Data is pushed to destination tables with Airbyte metadata columns

**2. Dual Storage Strategy** - Each micro-batch persists to /raw/<stream>/<YYYY-MM-DD>/<HH>/ paths - Enables complete replay capability and audit trails - Raw data also written to staging\_ods schema for immediate processing

**3. Webhook Notification** - Upon successful ingestion, Airbyte sends webhook to downstream consumers - Triggers NiFi processing workflows automatically - Ensures timely processing of new data batches

**4. Incremental Processing Integration** - NiFi reads only records newer than stored watermark - Uses \_airbyte\_extracted\_at timestamp for efficient incremental queries - Performs UPSERT operations into target warehouse schema - Advances watermark after each successful processing cycle

### Understanding Airbyte Metadata Columns

**Key Columns for Integration**

*-- Airbyte automatically adds these columns to synchronized tables*  
\_airbyte\_raw\_id *-- UUID: Unique identifier for each record*  
\_airbyte\_extracted\_at *-- TIMESTAMP WITH TIME ZONE: When record was extracted*  
\_airbyte\_generation\_id *-- BIGINT: Version/generation number*  
\_airbyte\_meta *-- JSONB: Additional metadata from Airbyte*

**Sample Data Structure**

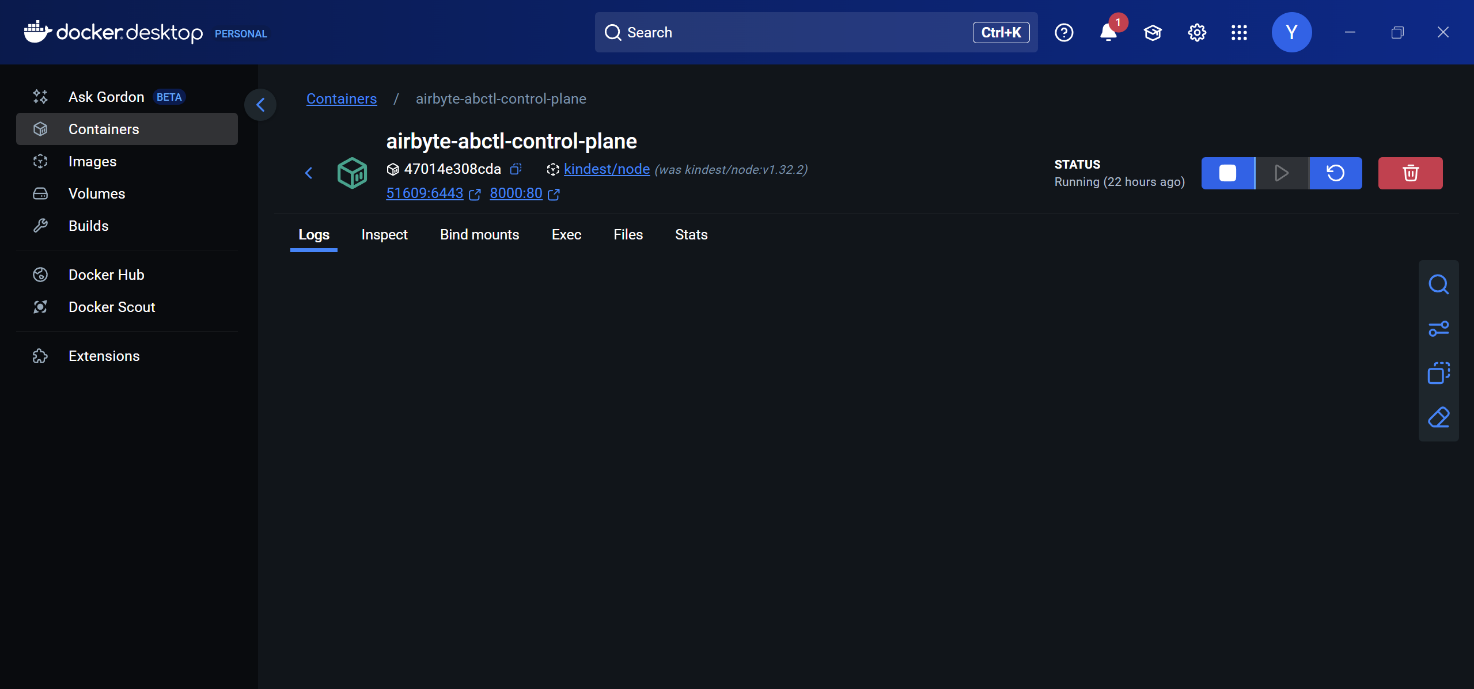
*-- Example data in scms\_data\_2 after Airbyte sync*  
**SELECT**   
 **id**,  
 name,  
 \_airbyte\_raw\_id,  
 \_airbyte\_extracted\_at,  
 \_airbyte\_generation\_id  
**FROM** **public**.scms\_data\_2  
**ORDER** **BY** \_airbyte\_extracted\_at **DESC**  
**LIMIT** 3;  
  
*-- Expected output:*  
*-- id | name | \_airbyte\_raw\_id | \_airbyte\_extracted\_at | generation\_id*  
*-- 1 | John | 123e4567-e89b-12d3-a456-426614174000 | 2025-08-31 17:30:00+05:30 | 0*  
*-- 2 | Jane | 123e4567-e89b-12d3-a456-426614174001 | 2025-08-31 17:30:00+05:30 | 0*  
*-- 3 | Bob | 123e4567-e89b-12d3-a456-426614174002 | 2025-08-31 17:30:00+05:30 | 0*

## Operational Guidance

### Starting the Pipeline

**Airbyte Setup on Windows**

1. **Start Docker Desktop** Airbyte relies on Docker, so ensure Docker Desktop is running in the background and check if the container is running.



1. **Open Command Prompt or PowerShell** Open a new Command Prompt or PowerShell window (not an older one—changes to environment variables only take effect in new windows).
2. **Navigate to Your Airbyte Directory** (if using Docker Compose)

* cd C:\path\to\airbyte

1. **Start Airbyte**
   * If using Docker Compose:
   * docker-compose up
   * If using abctl command-line utility:
   * abctl local install
   * For low-resource machines, add the --low-resource-mode flag:
   * abctl local install --low-resource-mode
2. **Open Airbyte in Your Browser**

* http://localhost:8000

**Tip**: If you have previously run the setup and only want to open Airbyte again, make sure Docker Desktop is running and then simply go to http://localhost:8000 in your browser.

### Post-deployment Checks

**Validation Steps**

* Confirm Airbyte run status in execution logs
* Check PostgreSQL staging\_ods schema for new data batches
* Verify raw storage partitions contain expected micro-batches
* Test webhook trigger functionality to downstream systems
* Validate schema synchronization and metadata column population

### Coordination with Downstream Processing

**Recommended Schedule Coordination**

Airbyte Sync Schedule: Every 6 hours (00:00, 06:00, 12:00, 18:00)  
NiFi Processing Schedule: Daily at 19:00 (after 18:00 Airbyte sync)  
  
Timeline Example:  
00:00 - Airbyte syncs batch #1  
06:00 - Airbyte syncs batch #2   
12:00 - Airbyte syncs batch #3  
18:00 - Airbyte syncs batch #4  
19:00 - NiFi processes all changes since last run (batches #1-4)

## Testing and Validation

### Initial Setup Validation

**Verify Database Connectivity**

*-- Test connection and data presence*  
**SELECT**   
 COUNT(\*) **as** total\_records,  
 MIN(\_airbyte\_extracted\_at) **as** earliest\_record,  
 MAX(\_airbyte\_extracted\_at) **as** latest\_record  
**FROM** **public**.scms\_data\_2;  
  
*-- Check target table initial state*  
**SELECT** COUNT(\*) **as** target\_count **FROM** **public**.scms\_data\_1;

**Validate Airbyte Configuration**

1. **Source Connection Test**:
   * Use “Test and save” button in source configuration
   * Verify successful connection to PostgreSQL
   * Confirm CDC publication and replication slot creation
2. **Destination Connection Test**:
   * Test destination PostgreSQL connection
   * Verify schema permissions and write access
   * Confirm proper SSL/security configuration

### Test Data Synchronization

**Add Test Data**

*-- Insert test records to verify CDC functionality*  
**INSERT** **INTO** source\_table (**id**, name, email, phone, address, created\_date)  
**VALUES**   
(9999, 'Test User 1', 'test1@example.com', '555-0001', '123 Test St', CURRENT\_DATE),  
(9998, 'Test User 2', 'test2@example.com', '555-0002', '456 Test Ave', CURRENT\_DATE);

**Monitor Sync Results**

1. **Check Airbyte Logs**:
   * Review sync execution logs for success/failure status
   * Verify record count matches expected values
   * Monitor for any CDC replication errors
2. **Validate Data Presence**:

* *-- Verify synchronized data in destination*  
  **SELECT** \* **FROM** **public**.scms\_data\_2   
  **WHERE** **id** **IN** (9999, 9998)  
  **ORDER** **BY** \_airbyte\_extracted\_at **DESC**;

### Webhook Testing

**Test Webhook Functionality**

1. **Configure webhook endpoint** in Airbyte connection settings
2. **Trigger manual sync** and monitor webhook delivery
3. **Verify downstream system** receives and processes webhook correctly

## Troubleshooting Guide

### Common Issues and Solutions

**Issue 1: CDC Replication Failures**

**Symptoms**: Airbyte fails to detect changes or sync stalls

**Solutions**:

1. **Check WAL configuration**:

* *-- Verify WAL level configuration*  
  SHOW wal\_level;  
  *-- Should be 'logical' for CDC*  
    
  *-- Check replication slot status*  
  **SELECT** \* **FROM** pg\_replication\_slots **WHERE** slot\_name = 'airbyte\_slot';

1. **Verify publication settings**:

* *-- Check publication exists and includes required tables*  
  **SELECT** \* **FROM** pg\_publication **WHERE** pubname = 'airbyte\_publication';  
  **SELECT** \* **FROM** pg\_publication\_tables **WHERE** pubname = 'airbyte\_publication';

**Issue 2: Connection Failures**

**Symptoms**: Airbyte cannot connect to PostgreSQL source or destination

**Solutions**:

1. **Validate connection parameters**:
   * Verify IP address, port, and database name
   * Check username/password credentials
   * Confirm firewall and network connectivity
2. **Test database connectivity**:

* *-- Test from Airbyte host machine*  
  psql -h 192.168.1.63 -p 5432 -U postgres -d scms\_db

**Issue 3: Schema Synchronization Issues**

**Symptoms**: Tables not appearing in schema discovery or sync failures

**Solutions**:

1. **Check table permissions**:

* *-- Verify user has necessary permissions*  
  **SELECT** grantee, privilege\_type   
  **FROM** information\_schema.role\_table\_grants   
  **WHERE** table\_name = 'scms\_data\_2';

1. **Refresh schema discovery**:
   * Click “Refresh source schema” in Airbyte UI
   * Verify table structure matches expected format
   * Check for special characters or reserved keywords in column names

**Issue 4: Performance and Timeout Issues**

**Symptoms**: Sync jobs timeout or perform poorly

**Solutions**:

1. **Adjust timeout settings**:
   * Increase “Initial Load Timeout” in advanced settings
   * Modify “Initial Waiting Time” for large datasets
   * Tune “Queue Size” based on available memory
2. **Optimize database performance**:

* *-- Add indexes for better performance*  
  **CREATE** **INDEX** CONCURRENTLY idx\_source\_timestamp   
  **ON** source\_table(updated\_at);  
    
  *-- Monitor query performance*  
  **SELECT** **query**, mean\_exec\_time, calls   
  **FROM** pg\_stat\_statements   
  **WHERE** **query** **LIKE** '%scms\_data%'  
  **ORDER** **BY** mean\_exec\_time **DESC**;

## Performance Optimization

### Airbyte Configuration Optimization

**Source Configuration Tuning**

* **Initial Waiting Time**: Set to 1200 seconds for large initial loads
* **Queue Size**: Configure to 10000 for optimal memory usage
* **LSN Commit Behaviour**: Use “After loading data in the destination” for consistency
* **Heartbeat Query**: Keep lightweight with SELECT 1

**Destination Configuration Tuning**

* **Batch Size**: Optimize based on target database capacity
* **Connection Pooling**: Configure appropriate pool sizes
* **Schema Caching**: Enable for better performance with stable schemas

### Database Optimization

**Index Strategy for Performance**

*-- Essential indexes for CDC performance*  
**CREATE** **INDEX** CONCURRENTLY idx\_scms\_data\_2\_extracted\_at   
**ON** **public**.scms\_data\_2(\_airbyte\_extracted\_at);  
  
**CREATE** **INDEX** CONCURRENTLY idx\_scms\_data\_2\_id   
**ON** **public**.scms\_data\_2(**id**);  
  
*-- Composite index for complex queries*  
**CREATE** **INDEX** CONCURRENTLY idx\_scms\_data\_2\_composite   
**ON** **public**.scms\_data\_2(\_airbyte\_extracted\_at, **id**);

**PostgreSQL Configuration for CDC**

*-- Optimize for replication workloads*  
**SET** wal\_level = 'logical';  
**SET** max\_replication\_slots = 10;  
**SET** max\_wal\_senders = 10;  
**SET** checkpoint\_completion\_target = 0.9;  
**SET** wal\_buffers = '16MB';

### Monitoring and Alerting

**Key Metrics to Monitor**

* **Airbyte Metrics**:
  + Sync success/failure rates
  + Record processing throughput
  + Sync duration and latency
  + Error rates and retry attempts
* **Database Metrics**:
  + Replication lag and slot usage
  + Connection count and wait times
  + WAL generation and consumption rates
  + Lock contention and blocking queries

**Performance Baselines**

Expected Performance (10,000 records):  
├── Initial Load: 5-15 minutes  
├── Incremental Sync: 1-3 minutes  
├── CDC Latency: < 5 seconds  
├── Memory Usage: 512MB-1GB  
└── Network Throughput: 10-50 MB/s

## Security and Best Practices

### Credential Management

**Secure Configuration**

* Store all credentials in environment variables or secure vaults
* Use dedicated service accounts with minimal required permissions
* Rotate passwords regularly and update Airbyte configurations
* Enable SSL/TLS for all database connections in production

### Access Control

**Role-Based Security**

*-- Create minimal privilege roles*  
**CREATE** **ROLE** airbyte\_reader;  
**GRANT** **USAGE** **ON** **SCHEMA** **public** **TO** airbyte\_reader;  
**GRANT** **SELECT** **ON** **ALL** **TABLES** **IN** **SCHEMA** **public** **TO** airbyte\_reader;  
  
**CREATE** **ROLE** airbyte\_writer;  
**GRANT** **USAGE** **ON** **SCHEMA** **public** **TO** airbyte\_writer;  
**GRANT** **INSERT**, **UPDATE** **ON** **ALL** **TABLES** **IN** **SCHEMA** **public** **TO** airbyte\_writer;

### Monitoring and Auditing

**Operational Best Practices**

* Monitor CDC replication lag and performance metrics
* Set up alerts for sync failures or performance degradation
* Regularly review Airbyte logs for security events
* Implement backup and recovery procedures for configuration
* Document all configuration changes and maintain version control

### Network Security

**Firewall and Network Configuration**

* Restrict database access to Airbyte host IP addresses
* Configure appropriate firewall rules for required ports
* Monitor network traffic for unusual patterns or unauthorized access