Healthcare Claims Database - Installation Guide for DBAs

Prerequisites

Required Privileges

- ACCOUNTADMIN or SECURITYADMIN role for security setup
- SYSADMIN role for database and schema creation
- CREATE DATABASE privilege
- CREATE WAREHOUSE privilege

System Requirements

- Snowflake Enterprise Edition (recommended for resource monitors)
- Minimum 100 credit allocation for demo environment
- Network access for demo users

Installation Steps

Step 1: Security and Access Setup

Script: `snowflake_security_setup.sql`
Run as: SECURITYADMIN or ACCOUNTADMIN
Duration: ~2 minutes

```sql
-- Execute the entire security setup script
-- This creates roles, users, warehouse, and resource monitors
```

Validates:

- Warehouse `healthcare_demo_wh` created
- 5 roles created (analyst, power_analyst, report_writer, data_steward, admin)
- 6 demo users created with passwords
- Resource monitor configured (100 credits/month)

Verification:

```sql
```

```
SHOW WAREHOUSES LIKE 'healthcare_demo_wh';
SHOW ROLES LIKE 'healthcare%';
SHOW USERS LIKE 'demo%';
```

Step 2: Database and Schema Creation

Script: `healthcare_claims_ddl.sql`
Run as: SYSADMIN or role with CREATE DATABASE privilege
Duration: ~1 minute

```sql
-- Create database if not exists
CREATE DATABASE IF NOT EXISTS healthcare_db;
USE DATABASE healthcare_db;

-- Execute DDL script to create all tables
-- Creates 17 tables with foreign keys and indexes
```

Validates:

- Database `healthcare_db` exists
- Schema `healthcare` created
- 17 tables created with proper relationships
- Indexes created for performance

Verification:

```sql
USE DATABASE healthcare_db;
USE SCHEMA healthcare;
SHOW TABLES;
SELECT COUNT(*) FROM INFORMATION_SCHEMA.TABLES
WHERE TABLE_SCHEMA = 'HEALTHCARE';
-- Expected: 17 tables
```

Step 3: Load Sample Data

Script: `healthcare_sample_data.sql`
Run as: Role with INSERT privilege (healthcare_demo_admin_role)
Duration: ~5 minutes

```sql
USE DATABASE healthcare_db;
USE SCHEMA healthcare;

-- Execute sample data script
-- Loads 100 claims with all related records
```

Validates:

- 20 patients loaded
- 100 claims with line items
- Payments, denials, appeals properly linked
- All foreign key relationships satisfied

Verification:

```sql
-- Run the verification query at end of sample data script
-- Should show counts for all 17 tables
SELECT 'Claims' as entity, COUNT(*) FROM claim
UNION ALL
SELECT 'Claim Line Items', COUNT(*) FROM claim_line_item;
-- Expected: 100 claims, ~400 line items
```

Step 4: Create Reporting Views

Script: `healthcare_reporting_views.sql`
Run as: Role with CREATE VIEW privilege
Duration: ~1 minute

```sql
USE DATABASE healthcare_db;
USE SCHEMA healthcare;

-- Execute views script
-- Creates 15 analytical views
```

Validates:

- 15 views created successfully
- Views accessible to analyst roles
- No errors in view definitions

Verification:

```sql
SHOW VIEWS IN SCHEMA healthcare;
SELECT COUNT(*) FROM INFORMATION_SCHEMA.VIEWS
WHERE TABLE_SCHEMA = 'HEALTHCARE';
-- Expected: 15 views

-- Test a view
SELECT * FROM v_claims_summary_dashboard LIMIT 10;
```

Step 5: Grant Permissions

Run as: SECURITYADMIN

```sql
-- Verify grants are in place
SHOW GRANTS TO ROLE healthcare_analyst_role;
SHOW GRANTS TO ROLE healthcare_power_analyst_role;

-- Grant future object privileges if needed
GRANT SELECT ON FUTURE TABLES IN SCHEMA healthcare_db.healthcare
TO ROLE healthcare_analyst_role;
GRANT SELECT ON FUTURE VIEWS IN SCHEMA healthcare_db.healthcare
TO ROLE healthcare_analyst_role;
```

Step 6: Test User Access

Run as: Each demo user role

```sql
```

```
-- Test as analyst
USE ROLE healthcare_analyst_role;
USE DATABASE healthcare_db;
USE SCHEMA healthcare;
SELECT COUNT(*) FROM claim;

-- Test as power analyst
USE ROLE healthcare_power_analyst_role;
CREATE TEMP TABLE test_temp AS SELECT * FROM claim LIMIT 10;
DROP TABLE test_temp;

-- Test as admin
USE ROLE healthcare_demo_admin_role;
SELECT * FROM demo_audit_log;
```

Post-Installation Tasks

Configure Monitoring

```sql
-- Enable query history tracking
ALTER SESSION SET USE_CACHED_RESULT = FALSE;

-- Monitor credit usage
SELECT * FROM SNOWFLAKE.ACCOUNT_USAGE.WAREHOUSE_METERING_HISTORY
WHERE WAREHOUSE_NAME = 'HEALTHCARE_DEMO_WH'
ORDER BY START_TIME DESC;
```

Set Up Scheduled Maintenance

```sql
-- Create task for regular statistics update (optional)
CREATE TASK IF NOT EXISTS update_table_stats
  WAREHOUSE = healthcare_demo_wh
  SCHEDULE = 'USING CRON 0 2 * * SUN America/Denver'
AS
  CALL SYSTEM$GATHER_STATS('healthcare_db.healthcare');
```

Configure Backup (Optional)

```sql
```

```
-- Enable Time Travel for recovery
ALTER DATABASE healthcare_db SET DATA_RETENTION_TIME_IN_DAYS = 7;

-- Create clone for backup
CREATE DATABASE healthcare_db_backup CLONE healthcare_db;
```

Troubleshooting

Common Issues and Solutions

Issue: Foreign key constraint violations during data load

```sql
-- Temporarily disable constraints
ALTER TABLE claim DROP CONSTRAINT <constraint_name>;
-- Load data
-- Re-enable constraints
ALTER TABLE claim ADD CONSTRAINT <constraint_name>
FOREIGN KEY (patient_id) REFERENCES patient(patient_id);
```

Issue: Insufficient privileges error

```sql
-- Check current role
SELECT CURRENT_ROLE();
-- Switch to appropriate role
USE ROLE SYSADMIN;
```

Issue: Warehouse suspended due to resource monitor

```sql
-- Check monitor status
SHOW RESOURCE MONITORS;
-- Temporarily increase limit or resume warehouse
ALTER RESOURCE MONITOR healthcare_demo_monitor
SET CREDIT_QUOTA = 150;
ALTER WAREHOUSE healthcare_demo_wh RESUME;
```

Issue: View creation fails due to missing columns

```sql
-- Verify base table structure
DESC TABLE claim;
-- Check for column name case sensitivity
SELECT * FROM claim LIMIT 1;
```

Validation Checklist

- [ ] All 17 tables created and populated
- [ ] All 15 views created without errors
- [ ] 6 demo users can login successfully
- [ ] Analyst role can SELECT from tables/views
- [ ] Power analyst can create temp tables
- [ ] Resource monitor active and configured
- [ ] Warehouse auto-suspend working (5 min idle)
- [ ] Sample queries return expected results

Rollback Procedure

If installation fails:

```sql
-- Drop in reverse order
DROP DATABASE IF EXISTS healthcare_db CASCADE;
DROP WAREHOUSE IF EXISTS healthcare_demo_wh;
DROP ROLE IF EXISTS healthcare_demo_admin_role;
DROP ROLE IF EXISTS healthcare_report_writer_role;
DROP ROLE IF EXISTS healthcare_power_analyst_role;
DROP ROLE IF EXISTS healthcare_data_steward_role;
DROP ROLE IF EXISTS healthcare_analyst_role;
DROP USER IF EXISTS demo_analyst_1;
-- Continue for all demo users...
DROP RESOURCE MONITOR IF EXISTS healthcare_demo_monitor;
```

Performance Optimization

Recommended Settings

```sql
```

```sql
-- Set warehouse size based on usage
ALTER WAREHOUSE healthcare_demo_wh
SET WAREHOUSE_SIZE = 'MEDIUM' -- For larger demos

-- Enable query acceleration
ALTER WAREHOUSE healthcare_demo_wh
SET ENABLE_QUERY_ACCELERATION = TRUE;

-- Set appropriate clustering keys
ALTER TABLE claim CLUSTER BY (submission_date, claim_status);
ALTER TABLE claim_line_item CLUSTER BY (claim_id);
```

Security Hardening

For production use:

```sql
-- Enable MFA for admin roles
ALTER USER demo_admin SET MINS_TO_BYPASS_MFA = 0;

-- Set session timeout
ALTER USER demo_analyst_1 SET SESSION_IDLE_TIMEOUT_MINS = 30;

-- Enable audit logging
CREATE OR REPLACE TABLE access_log AS
SELECT * FROM SNOWFLAKE.ACCOUNT_USAGE.QUERY_HISTORY
WHERE DATABASE_NAME = 'HEALTHCARE_DB';
```

Contact Information

- **Installation Issues:** Contact DBA team
- **Access Requests:** Submit via ticketing system
- **Performance Issues:** Monitor warehouse utilization first
- **Data Issues:** Check demo_audit_log table

Script Execution Order Summary

1. `snowflake_security_setup.sql` (SECURITYADMIN)
2. `healthcare_claims_ddl.sql` (SYSADMIN)
3. `healthcare_sample_data.sql` (healthcare_demo_admin_role)

4. `healthcare_reporting_views.sql` (healthcare_demo_admin_role)

Total estimated installation time: ~10 minutes