Datasets Module How-to

Manage databases, tables, and queries as reusable datasets.

How-to The Four Pillars How-to Business Operations How-to Datasets | Tutorial | How-to Guide | Reference

---

Overview

Datasets Module (How-to Guide) walks you through configuring the Datasets module for bi-directional database communication. You'll create database connections, build queries, map tables to tags, and integrate query results with displays and scripts.

Prerequisites:

- Database connection information
- Understanding of SQL syntax
- Tags created for data mapping (optional)

Configuration Workflow
Step 1: Define Database Connections
Step 2: Create Queries
Step 3: Map Database Tables
Step 4: Configure Files (Optional)
Step 5: Test and Display Data
Runtime Execution
Common Issues
Best Practices Checklist
Security Considerations
Datasets Module Links

Configuration Workflow

1. **Define Database Connections** - Configure DatasetDB objects
2. **Create Queries** - Build SQL statements with parameters
3. **Map Tables** (Optional) - Link database tables to tags
4. **Configure Files** (Optional) - Setup text/XML file I/O
5. **Test and Display** - Verify queries and show results

Step 1: Define Database Connections

Using Default SQLite Databases

Every solution includes four pre-configured SQLite databases:

Database	Purpose	File Location
TagHistorian	Time-series data	`<SolutionName>.dbTagHistorian`
AlarmHistorian	Alarm events & audit	`<SolutionName>.dbAlarmHistorian`
Retentive	Persistent tag values	`<SolutionName>.dbRetentive`
RuntimeUsers	Dynamic users	`<SolutionName>.dbRuntimeUsers`

Adding External Database

1. Navigate to **Datasets DBs**
2. Click **Plus** icon
3. Configure connection:
    - **Name**: Database identifier
    - **Provider**: Connection type
    - **Database**: Database system
    - **Description**: Documentation
4. Click **OK**
5. Configure ConnectionString properties

Connection Examples

Database	Provider	Connection String
SQL Server	SqlClient	`Server=.\SQLEXPRESS;Database=MyDB;`
PostgreSQL	PostgreSQL	`Server=localhost;Database=MyDB;Port=5432;`
MySQL	ODBC	`Driver={MySQL};Server=localhost;Database=MyDB;`
Oracle	OracleClient	`Data Source=//localhost:1521/XE;`

Testing Connection

1. After configuring, click **Test** button
2. Provide credentials if required
3. Verify successful connection

Step 2: Create Queries

Building a Query

1. Go to **Datasets Queries**
2. Click **Plus** icon
3. Enter **Name** (no spaces, use underscores)
4. Select **DB** from dropdown
5. Click **OK**
6. Double-click **SqlStatement** to open editor

SQL Statement with Parameters

Use curly brackets to embed tag values:

Static Query:

sql

```sql
SELECT * FROM Products WHERE Price > 100
```

Dynamic Query with Tags:

sql

```sql
SELECT * FROM Products
WHERE Price > {{Tag.MinPrice}}
AND Category = '{{Tag.Category}}'
```

Query Editor Features

- Syntax highlighting
- IntelliSense support
- Parameter validation
- Test execution

Visual Query Builder

For complex queries without SQL knowledge:

1. Go to **Datasets DBs**
2. Click **Visual Query Builder**
3. Drag tables and connect fields visually

Step 3: Map Database Tables

Creating Table Mapping

1. Navigate to **Datasets Tables**
2. Click **Plus** icon
3. Configure:
    - **Name**: Table identifier
    - **DB**: Database connection
    - **TableName**: Actual database table
    - **WhereCondition**: Filter criteria
    - **Access**: Read/Insert/ReadWrite
4. Click **OK**

Mapping Tags to Columns

1. Select table in grid
2. Click **Mapping** button (...)
3. Map database columns to tags:
    - Left: Database columns

    - Right: Solution tags
4. Configure **DateTimeMode** (UTC/Local)

Step 4: Configure Files (Optional)

Setting Up File I/O

1. Go to **Datasets Files**
2. Click **Plus** icon
3. Configure:
    - **Name**: File identifier
    - **FileName**: Path with optional tag values
    - **FileType**: ASCII/Unicode/XML
    - **Objects**: Tags to read/write
4. Click **OK**

Dynamic File Names

Use tags in file paths:

```
C:\Recipes\Recipe_{{Tag.RecipeID}}.xml
```

Step 5: Test and Display Data

Testing Queries

1. Go to **DataExplorer SQL Queries**
2. Select database and query
3. Click **Execute**
4. Review results in grid

Displaying in Runtime

Using DataGrid Control:

1. In **Displays Draw**
2. Add **DataGrid** control
3. Set DataSource property:
   `Dataset.Query.QueryName`

Using Scripts:

csharp

```csharp
// Execute query
DataTable result = @Dataset.Query.MyQuery.SelectCommand();

// Process results
foreach(DataRow row in result.Rows)
{
    @Tag.Value = row["ColumnName"];
}
```

Runtime Execution

Synchronous vs Asynchronous

Synchronous - Blocks until complete:

csharp

```csharp
DataTable dt = @Dataset.Query.MyQuery.SelectCommand();
```

Asynchronous - Non-blocking:

csharp

```csharp
@Dataset.Query.MyQuery.ExecuteAsync();
// Check LastStatus property for completion
```

Store and Forward

Store and Forward applies only to the Tag Historian and Alarm history databases, not to other Dataset database connections:

1. Enable it in Historian > Storage Locations for the Tag Historian, or in Alarms > Global Settings for alarm history
2. System buffers data during connection loss
3. Automatic forward when connection restored

Common Issues

Connection Failed

- Verify server address and port
- Check firewall settings
- Confirm credentials
- Test network connectivity

Query Syntax Error

- Validate SQL syntax for your database type
- Check table and column names
- Verify data types in WHERE conditions
- Test in SQL tool first

No Data Returned

- Check WHERE conditions
- Verify table has data
- Confirm user permissions
- Review date/time filters

Performance Issues

- Add database indexes
- Limit result sets (TOP/LIMIT)
- Use stored procedures
- Enable connection pooling

Best Practices Checklist

- **Use parameterized queries** - Prevent SQL injection with tag parameters
- **Test in DataExplorer** - Verify queries before runtime use
- **Handle errors** - Check LastStatus property
- **Limit result sets** - Use WHERE and LIMIT clauses
- **Document connections** - Use description fields
- **Secure credentials** - Only admins set passwords
- **Plan database schema** - Design before implementation

Security Considerations

FrameworX provides two built-in approaches to prevent SQL Injection when interacting with databases. The choice depends on the use case and complexity of the operation.

Approach 1: Dataset Tables

Dataset Tables automatically generate INSERT and UPDATE commands using internal ADO parameterized queries. Since the framework builds these operations internally with typed parameters, tag values are never concatenated into the SQL string. SQL Injection protection is automatic and transparent.

This approach is recommended for straightforward insert and update operations that do not require custom database logic.

Approach 2: Stored Procedures / Functions via the `execute` command

When the `execute` keyword is used in a Dataset Query, FrameworX parses the command internally and calls the Stored Procedure (SQL Server) or Function (PostgreSQL) through ADO with typed parameter passing. Values are sent as ADO parameters — never concatenated into the SQL command — eliminating the risk of SQL Injection.

For detailed syntax, examples, and configuration, see DatasetQuery Stored Procedure

Summary

Scenario	SQL Injection Risk	Recommendation
Hardcoded query, no tags	No risk — no user input involved	Direct query is safe
Query with tags not exposed to the user	No risk — values are controlled internally	Direct query is safe
Query with tags exposed to user input	**Yes — user can manipulate values**	Dataset Tables or `execute` with Stored Procedures /Functions

Network Gateway

For routing database access across network security layers:

1. Set GatewayIP in DB configuration
2. Install TWebServices on gateway machine
3. Route through secure connection

Datasets Module Links

Explanation - to understand concepts

Modules / Business Operations / Datasets Module

Tutorials - to learn by doing

Tutorials / Business Operations / Datasets Module Tutorial

How-to Guides - to accomplish specific tasks

How-to Guides / Business Operations / Datasets Module How-to Guide

Reference - technical details

Technical Reference / Business Operations / Datasets Module Reference

---

In this section...
