Northwind BI solution

Denis Zinukov

[@AzureDevOps](https://dev.azure.com/zinykov/20767/)

[@VK](https://vk.com/denis.zinukov)

[@Linkedin](http://www.linkedin.com/in/deniszinukov)

Table of contents

[Introduction 5](#_Toc185084014)

[Planning a BI Solution 6](#_Toc185084015)

[High-level project schema 6](#_Toc185084016)

[Project workflow 7](#_Toc185084017)

[Project Scope 8](#_Toc185084018)

[Required BI Software 8](#_Toc185084019)

[Project Infrastructure 8](#_Toc185084020)

[Stakeholders 8](#_Toc185084021)

[Personnel roles 8](#_Toc185084022)

[Project Management Infrastructure 9](#_Toc185084023)

[Design and Development Tools 9](#_Toc185084024)

[Test infrastructure 10](#_Toc185084025)

[Planning SQL Server Business Intelligence Infrastructure 11](#_Toc185084026)

[BI Topology 11](#_Toc185084027)

[Single BI server 11](#_Toc185084028)

[Dedicated report server 12](#_Toc185084029)

[Dedicated ETL server 13](#_Toc185084030)

[Dedicated master data server 14](#_Toc185084031)

[Distributed report server 15](#_Toc185084032)

[Distributed master data server 16](#_Toc185084033)

[Dedicated landing server 17](#_Toc185084034)

[Planning Data Warehouse Hardware 18](#_Toc185084035)

[Designing Data Warehouse 19](#_Toc185084036)

[Designing Data Warehouse Logical Schema 19](#_Toc185084037)

[Identify business processes 19](#_Toc185084038)

[Designing Dimension Models and Data Warehouse Tables 20](#_Toc185084039)

[Dimension tables definition 21](#_Toc185084040)

[Facts tables definition 23](#_Toc185084041)

[Designing Data Warehouse Physical Implementation 24](#_Toc185084042)

[Designing File Storage 24](#_Toc185084043)

[Partitioning 24](#_Toc185084044)

[Indexes 24](#_Toc185084045)

[Compression 25](#_Toc185084046)

[Views 25](#_Toc185084047)

[Designing ETL Solution 26](#_Toc185084048)

[ETL Architecture 26](#_Toc185084049)

[Planning Data Extraction 27](#_Toc185084050)

[Data source profiling 27](#_Toc185084051)

[Identifying New and Modified Rows 27](#_Toc185084052)

[Planning Extraction Windows 28](#_Toc185084053)

[Planning Data Transformation 29](#_Toc185084054)

[Data Flow Diagrams 29](#_Toc185084055)

[Dimensions 29](#_Toc185084056)

[Facts 34](#_Toc185084057)

[Source To Target Mapping 35](#_Toc185084058)

[Handling Errors 36](#_Toc185084059)

[Integration Services (SSIS) Logging 36](#_Toc185084060)

[Designing BI Delivery Solution 37](#_Toc185084061)

[Reporting Requirements 37](#_Toc185084062)

[Development operations 38](#_Toc185084063)

[Solution artifacts 38](#_Toc185084064)

[Solution environments 38](#_Toc185084065)

[Configuring debug environment 38](#_Toc185084066)

[Unit tests 39](#_Toc185084067)

[Functional ETL test 39](#_Toc185084068)

[Continuous integration/continuous deployment 40](#_Toc185084069)

[Operation of BI system 41](#_Toc185084070)

[ETL Operations 41](#_Toc185084071)

[Execution context 41](#_Toc185084072)

[Execution account 41](#_Toc185084073)

[Data Warehouse Operations 42](#_Toc185084074)

[Reorganizing or rebuilding indexes 42](#_Toc185084075)

[Updating data distribution statistics 42](#_Toc185084076)

[Managing partitioned tables 42](#_Toc185084077)

[Security 42](#_Toc185084078)

[DQS Operations 42](#_Toc185084079)

[Configuration 42](#_Toc185084080)

[Security 43](#_Toc185084081)

[MDS Operations 43](#_Toc185084082)

[Security 43](#_Toc185084083)

[Cleaning up transactions and staging tables 43](#_Toc185084084)

[Reporting Services Operations 44](#_Toc185084085)

[Managing Scheduled Tasks 44](#_Toc185084086)

[Backing up BI Solution 44](#_Toc185084087)

[Manual Operational Tasks 45](#_Toc185084088)

[Managing database files 45](#_Toc185084089)

[Automating Operational Tasks 45](#_Toc185084090)

[SQL agent jobs 45](#_Toc185084091)

[Maintenance plan: NorthwindBI 46](#_Toc185084092)

[Maintenance subplan: DWH Maintenance 46](#_Toc185084093)

[Maintenance subplan: Full backup 47](#_Toc185084094)

[Maintenance subplan: Differential backup 47](#_Toc185084095)

[Maintenance subplan: Log backup 48](#_Toc185084096)

[ETL execution duration 48](#_Toc185084097)

[Disaster recovery plan 48](#_Toc185084098)

[Monitoring and Optimizing BI Solution 49](#_Toc185084099)

[Creating a Performance Baseline 49](#_Toc185084100)

[Monitoring ETL 49](#_Toc185084101)

[Package Execution Reports 49](#_Toc185084102)

[Integration.Lineage 49](#_Toc185084103)

[Monitoring and optimizing Data Warehouse 49](#_Toc185084104)

[Considerations for Data Warehouse Performance 49](#_Toc185084105)

[Data warehouse workloads: 49](#_Toc185084106)

[SQL Server settings 50](#_Toc185084107)

[Monitoring tools 50](#_Toc185084108)

[Turning tools 50](#_Toc185084109)

[Using Resource Governor to Balance Resource Utilization 50](#_Toc185084110)

[Troubleshooting Data Warehouse Performance 51](#_Toc185084111)

[Monitoring DQS 51](#_Toc185084112)

[Monitoring MDS 51](#_Toc185084113)

[Monitoring and optimizing Power BI Report Server 52](#_Toc185084114)

[Considerations for Reporting Services Performance 52](#_Toc185084115)

[Reporting Services workloads 52](#_Toc185084116)

[Reporting Services settings 52](#_Toc185084117)

[Monitoring Tools 52](#_Toc185084118)

[Tune 52](#_Toc185084119)

[Using Caching and Snapshots to Optimize Performance 52](#_Toc185084120)

# Introduction

Northwind Traders is a fictitious organization that manages orders, products, customers, suppliers, and many other aspects of a small business. This example is distributed by Microsoft.

The goal of this project is to develop a BI solution that provides sales analytics for the management of Northwind Traders.

# Planning a BI Solution

## High-level project schema

Data sources

Structured



Semi-structured

ETL

*SQL Server Integration Services*



Reporting & analysis

*Power BI Report Server*

Data Cleansing

*SQL Server*

*Data Quality Services*



Master data

management

*SQL Server*

*Master Data Services*



Data

Warehouse

*SQL Server*

*Database engine*



## Project workflow[[1]](#footnote-2)

Project Planning

Business requirements definition

Technical architecture design

Dimensional modelling

BI application design

Product selection & installation

Physical design

ETL design & development

BI application development

Deployment

Maintenance

Growth

Project management

## Project Scope

|  |  |  |
| --- | --- | --- |
| High business value |  | * See sales amount with discount by customers and products. * Compare sales amount with discount with previous year/quarter/month. * See discount by employees and their geography. * See new and returned customers. * Cluster customers based on spending volume. * Product ABC classification. * Product basket analysis. * See number of orders, sales amount, sales amount with discount in progress. |
| Low business value |  |  |
|  | Low feasibility | High feasibility |

## Required BI Software

|  |  |
| --- | --- |
| Software | Rationale |
| SQL Server Database Engine | * The CEO wants a consistent view of all business data, and a centralized data warehouse in SQL Server would provide this. |
| SQL Server Integration Services | * The business data required for analysis and reporting is currently spread across a range of data sources. Integration Services will provide an ETL platform to populate and refresh the data warehouse. |
| SQL Server Master Data Services | * The Sales VP has complained about inconsistent data, which could potentially be caused by a lack of central data management for key business entities. |
| SQL Server Data Quality Services | * The specialists have difficulty ensuring that their analysis of sales data is accurate because of quality issues in the data. Data Quality Services could be used to cleanse records and improve the consistency, thereby reducing the inaccuracy of the sales analysis. |
| Power BI Report Server | * Business users need the centralized portal for analysis and reporting. * The database administrators team needs to receive email notifications about errors in the ETL process. * Data stewards need to be notified of errors and raw values in data cleansing processes. |

## Project Infrastructure

### Stakeholders

* **Executive sponsor.** Ensures BI project supports business vision. Crucial for project success.
* **Business users.** Provide detailed understanding of business processes, terminology, and metrics.
* **Data stewards.** Use business knowledge to validate and maintain business entity data.

### Personnel roles

* **A project manager.** Coordinates project tasks and schedules and ensures that the project is completed on time and within budget.
* **A BI solution architect**. Has overall responsibility for the technical design of the data warehousing solution.
* **A system analyst.** Examines and understands the data-related requirements, objectives, and constraints of the organization to design and develop an effective BI system.
* **A data engineer.** Builds database and ETL workflows to meet business requirements in analysis and reporting.
* **A data analyst.** Creates the reporting elements of the BI solution.
* **A DevOps specialist.** Implements server and network infrastructure for a data warehouse solution, migrates code between environments, sets up CI/CD pipelines.
* **A Database administrator.** Maintains data warehouse, provides access for users.
* **Business users.** Provide requirements and help to prioritize the business questions that the data warehousing solution will answer. Often, the team includes a business analyst as a full-time member to help interpret the business questions and ensure that the solution design meets the needs of the users.
* **Testers.** Verify the business and operational functionality of the solution as it is developed.

### Project Management Infrastructure

* Microsoft Office
* Microsoft Project
* A SharePoint Server site for the project.

### Design and Development Tools

* Microsoft Visio
* [SQL Server Data Tools](https://learn.microsoft.com/en-us/sql/ssdt/download-sql-server-data-tools-ssdt) & [Extensions](https://learn.microsoft.com/en-us/sql/ssdt/download-sql-server-data-tools-ssdt#install-extensions-for-analysis-services-integration-services-and-reporting-services)
* [Microsoft Power BI Desktop (Optimized for Power BI Report Server)](https://learn.microsoft.com/en-us/power-bi/report-server/install-powerbi-desktop)
* [Tabular Editor](https://sql.bi/53569)
* [DAX Studio](https://sql.bi/980)
* [Analyze in Excel for Power BI Desktop](https://sql.bi/675172)
* [ALM Toolkit](https://sql.bi/629988)
* Azure DevOps (Server or Service)

### 

### Test infrastructure

Table 1. Test environment hardware & software

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Server | CPU, cores | RAM, GB | Storage, GB | Software | Collation |
| Test Server | 8 | 16 | 20 | * SQL Server 2022 Developer Edition   + SQL Server DB Engine   + SQL Server Integration Services   + Data Quality Services   + Master Data Services * Power BI Report Server | SQL\_Latin1\_General\_CP1\_CI\_AS |

Table 2. Test server file storage

|  |  |  |  |
| --- | --- | --- | --- |
| Logical Drive | Type | Storage, GB | Databases & Filegroups |
| Disk1 | SSD | 20 | 1. Data Warehouse 2. Landing zone 3. Staging databases 4. SSIS catalog 5. Reporting Services catalog 6. DQS catalog 7. MDS catalog 8. Logs database |

# Planning SQL Server Business Intelligence Infrastructure

## BI Topology

### Single BI server



* SQL Server DB Engine
  + Data Warehouse
  + Landing zone
  + Staging databases
  + SSIS catalog
  + Reporting Services catalog
  + DQS catalog
  + MDS catalog
  + Logs database
* SQL Server Integration Services
* Power BI Report Server
* Data Quality Services
* Master Data Services

SRVBI



CLIENT PC

* Microsoft Excel
  + Power Query
  + Power Pivot
  + Power View
* Report Builder
* Power BI Desktop
* Microsoft Edge

Notes:

* Minimal server hardware and software license requirements, but the server would require significant memory, CPU, and disk resources for all but the most lightweight BI workloads.
* The range of different workload types on the server would make it difficult to specify and configure hardware resources appropriately.
* The server could be clustered to provide high availability.

### 

### Dedicated report server



* SQL Server DB Engine
  + Data Warehouse
  + Landing zone
  + SSIS catalog
* SQL Server Integration Services

SRVDWH



CLIENT PC

* Microsoft Excel
  + Power Query
  + Power Pivot
  + Power View
* Report Builder
* Power BI Desktop
* Microsoft Edge



* SQL Server DB Engine
  + Reporting Services catalog
  + DQS catalog
  + MDS catalog
  + Logs database
  + Staging databases
* Power BI Report Server
* Data Quality Services
* Master Data Services

SRVPBIRS

Notes:

* The reporting, DQS and MDS and DWH workloads are separated.
* Using a dedicated reporting, DQS and MDS server makes it easier to manage different types of workloads.
* Additional report servers could be added to scale out reporting.
* One or both servers could be clustered to provide high availability.

### 

### Dedicated ETL server



* SQL Server DB Engine
  + Data Warehouse
  + Landing zone

SRVDWH



CLIENT PC

* Microsoft Excel
  + Power Query
  + Power Pivot
  + Power View
* Report Builder
* Power BI Desktop
* Microsoft Edge



* SQL Server DB Engine
  + Reporting Services catalog
  + DQS catalog
  + MDS catalog
  + Logs database
  + Staging databases
* Power BI Report Server
* Data Quality Services
* Master Data Services

SRVPBIRS



* SQL Server DB Engine
  + SSIS catalog
* SQL Server Integration Services

SRVETL

Notes:

* The DWH, reporting and ETL workloads are full separated.
* The landing database could be hosted on the ETL server or data warehouse server depending on workloads.
* SRVDWH, SRVPBIRS, and SRVETL could be clustered to provide high availability.
* Additional report servers and ETL could be added to scale out.

### 

### Dedicated master data server



* SQL Server DB Engine
  + Data Warehouse
  + Landing zone

SRVDWH



CLIENT PC

* Microsoft Excel
  + Power Query
  + Power Pivot
  + Power View
* Report Builder
* Power BI Desktop
* Microsoft Edge



* SQL Server DB Engine
  + Reporting Services catalog
* Power BI Report Server

SRVPBIRS



* SQL Server DB Engine
  + DQS catalog
  + MDS catalog
  + Staging databases
  + Logs database
* Data Quality Services
* Master Data Services

SRVMDM



* SQL Server DB Engine
  + SSIS catalog
* SQL Server Integration Services

SRVETL

Notes:

* The DWH, reporting, ETL and MDM workloads are full separated.
* The landing database could be hosted on the ETL server or data warehouse server depending on workloads.
* Servers could be clustered to provide high availability.
* Additional report servers and ETL could be added to scale out.

### 

### Distributed report server



* SQL Server DB Engine
  + Data Warehouse
  + Landing zone

SRVDWH



CLIENT PC

* Microsoft Excel
  + Power Query
  + Power Pivot
  + Power View
* Report Builder
* Power BI Desktop
* Microsoft Edge



* Power BI Report Server

SRVPBIRS



* SQL Server DB Engine
  + DQS catalog
  + MDS catalog
  + Logs database
  + Staging databases
* Data Quality Services
* Master Data Services

SRVMDM



* SQL Server DB Engine
  + SSIS catalog
* SQL Server Integration Services

SRVETL



* SQL Server DB Engine
  + Reporting Services catalog

SRVPBIRSDB

Notes:

* The DWH, reporting, ETL and MDM workloads are full separated.
* The landing database could be hosted on the ETL server or data warehouse server depending on workloads.
* The report server uses a separate database server to host the report catalog.
* Servers could be clustered to provide high availability.
* Additional report servers and ETL could be added to scale out.

### Distributed master data server



* SQL Server DB Engine
  + Data Warehouse
  + Landing zone

SRVDWH



CLIENT PC

* Microsoft Excel
  + Power Query
  + Power Pivot
  + Power View
* Report Builder
* Power BI Desktop
* Microsoft Edge



* Power BI Report Server

SRVPBIRS



* Data Quality Services
* Master Data Services

SRVMDM



* SQL Server DB Engine
  + SSIS catalog
* SQL Server Integration Services

SRVETL



* SQL Server DB Engine
  + Reporting Services catalog

SRVPBIRSDB



* SQL Server DB Engine
  + DQS catalog
  + MDS catalog
  + Logs database
  + Staging databases

SRVMDMDB

Notes:

* The DWH, reporting, ETL and MDM workloads are full separated.
* The landing database could be hosted on the ETL server or data warehouse server depending on workloads.
* The report server uses a separate database server to host the report catalog.
* Servers could be clustered to provide high availability.
* Additional report servers and ETL could be added to scale out.

### Dedicated landing server



* SQL Server DB Engine
  + Data Warehouse

SRVDWH



CLIENT PC

* Microsoft Excel
  + Power Query
  + Power Pivot
  + Power View
* Report Builder
* Power BI Desktop
* Microsoft Edge



* Power BI Report Server

SRVPBIRS



* Data Quality Services
* Master Data Services

SRVMDM



* SQL Server DB Engine
  + SSIS catalog
* SQL Server Integration Services

SRVETL



* SQL Server DB Engine
  + Reporting Services catalog

SRVPBIRSDB



* SQL Server DB Engine
  + DQS catalog
  + MDS catalog
  + Logs database
  + Staging databases

SRVMDMDB



* SQL Server DB Engine
  + Landing zone

SRVLDN

Notes:

* The DWH, reporting, ETL, MDM and landing workloads are full separated.
* The landing database could be hosted on the ETL server or data warehouse server depending on workloads.
* The report server uses a separate database server to host the report catalog.
* Servers could be clustered to provide high availability.
* Additional report servers and ETL could be added to scale out.

## Planning Data Warehouse Hardware

[DW Hardware Spec.xlsx](https://dev.azure.com/zinykov/20767/_git/Northwind_BI_Solution?path=/Docs/DW%20Hardware%20Spec.xlsx&version=GBmaster&_a=contents)

# Designing Data Warehouse

## Designing Data Warehouse Logical Schema

### Identify business processes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Business processes** | **Dimensions** | | | |
| Date | Product | Customer | Employee |
| Orders | X | X | X | X |

Notes:

* There is no interest in shipment geography, so there is no reason to create dedicated geography dimension. Employee and Customer geography will be implemented as attributes.
* In Date dimension will be used opensource project [Производственный календарь в XML (xmlcalendar.ru)](http://xmlcalendar.ru/) for getting information about production calendar.

### Designing Dimension Models and Data Warehouse Tables

Figure 1 - Order dimensional model

Date

(Order, Required, Shipped)

Year

Quarter

Month

Day

Year

Quarter

Mon

ISO Week

Weekday

Year

Quarter

Month

Week

Weekday

Customer

Country

City

Customer

Contact name

Contact title

Phone

Fax

DayOfQuarterNumber

DayOfYearNumber

StartOfYear

EndOfYear

YearQuarter

StartOfQuarter

EndOfQuarter

YearMonth

StartOfMonth

EndOfMonth

StartOfWeek

EndOfWeek

Holiday

Workday type

Workday hours

Product

Category

Product

Employee

Country

City

Employee

Title

Title of courtesy

### Dimension tables definition

Table 3 - Customer dimension

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| COLUMN NAME | COLUMN TYPE | SCD TYPE | DATA TYPE | NULL | Description |
| CustomerKey | Surrogate key |  | INT | NOT | Primary Key in DWH |
| CustomerAlterKey | Business key |  | NVARCHAR(5) | NOT | Primary key in Northwind (упрощённая) |
| Customer | Drill-thought detail | 1 | NVARCHAR(50) | NOT | Customer company name |
| ContactName | Drill-thought detail | 1 | NVARCHAR(50) |  | Customer representative |
| ContactTitle | Slicer | 1 | NVARCHAR(50) |  | Position of customer representative |
| Country | Hierarchy | 2 | NVARCHAR(25) |  | Customer country |
| City | Hierarchy | 2 | NVARCHAR(25) |  | Customer city |
| Phone | Drill-thought detail | 1 | NVARCHAR(30) |  | Customer phone |
| Fax | Drill-thought detail | 1 | NVARCHAR(30) |  | Customer fax |
| AllAttributes | Metadata | 1 | NVARCHAR(MAX) | NOT | JSON with all data about member |
| StartDate | Metadata |  | DATETIME2 | NOT | The time point from which the row is active |
| EndDate | Metadata |  | DATETIME2 |  | The point in time until which the row is active |
| Current | Metadata |  | BIT | NOT | Mark, that the row is active |
| LineageKey | Metadata |  | INT | NOT | Foreign key to audit table |

Table 4 - Date dimension

| COLUMN NAME | COLUMN TYPE | SCD TYPE | DATA TYPE | NULL | Description |
| --- | --- | --- | --- | --- | --- |
| DateKey | Surrogate key |  | DATE | NOT | Primary Key in DWH |
| DayOfMonth | Hierarchy |  | TINYINT | NOT | Day number of the month |
| DayOfWeek | Hierarchy |  | NVARCHAR(5) | NOT | Day name of the week |
| DayOfWeekNumber | Sort by column |  | TINYINT | NOT | Day number of the week |
| DayOfQuarterNumber | Drill-thought detail |  | TINYINT | NOT | Day number of the quarter |
| DayOfYearNumber | Drill-thought detail |  | SMALLINT | NOT | Day number of the year |
| Year | Hierarchy |  | SMALLINT | NOT | Year number |
| StartOfYear | Drill-thought detail |  | DATE | NOT | First date of year |
| EndOfYear | Drill-thought detail |  | DATE | NOT | Last date of year |
| Quarter | Hierarchy |  | NVARCHAR(5) | NOT | Quarter name |
| YearQuarter | Hierarchy |  | NVARCHAR(10) | NOT | Year + quarter name |
| YearQuarterNumber | Sort by column |  | INT | NOT | Year + quarter number |
| StartOfQuarter | Drill-thought detail |  | DATE | NOT | First date of quarter |
| EndOfQuarter | Drill-thought detail |  | DATE | NOT | Last date of quarter |
| Month | Hierarchy |  | NVARCHAR(10) | NOT | Month name |
| Mon | Hierarchy |  | NVARCHAR(5) | NOT | Short month name |
| MonthNumber | Sort by column |  | TINYINT | NOT | Month number of the year |
| YearMonth | Hierarchy |  | NVARCHAR(10) | NOT | Year + month name |
| StartOfMonth | Drill-thought detail |  | DATE | NOT | First date of month |
| EndOfMonth | Drill-thought detail |  | DATE | NOT | Last date of month |
| Week | Hierarchy |  | NVARCHAR(50) | NOT | Week name in format first date of week – last date of week |
| IOSWeekNumber | Hierarchy |  | TINYINT | NOT | Week number of the year according to ISO 8601 |
| StartOfWeek | Drill-thought detail |  | DATE | NOT | First date of week |
| EndOfWeek | Drill-thought detail |  | DATE | NOT | Last date of week |
| Holiday | Drill-thought detail |  | NVARCHAR(100) | NOT | Holiday name. Russian holidays |
| WorkDayType | Slicer |  | NVARCHAR(25) | NOT | Work/Short/ Day off |
| WorkDayHours | Measure |  | TINYINT | NOT | Number of hours in a work shift |

Table 5 - Employee dimension

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| COLUMN NAME | COLUMN TYPE | SCD TYPE | DATA TYPE | NULL | Description |
| EmployeeKey | Surrogate key |  | INT | NOT | Primary Key in DWH |
| EmployeeAlterKey | Business key |  | INT | NOT | Primary key in Northwind (упрощённая) |
| Employee | Drill-thought detail | 1 | NVARCHAR(35) | NOT | First name and last name of Employee |
| Title | Drill-thought detail | 1 | NVARCHAR(30) |  | Position |
| TitleOfCourtesy | Drill-thought detail | 1 | NVARCHAR(10) |  | Appeal to |
| City | Hierarchy | 2 | NVARCHAR(25) |  | City of residence of the employee |
| Country | Hierarchy | 2 | NVARCHAR(25) |  | Country of residence of the employee |
| AllAttributes | Metadata | 1 | NVARCHAR(MAX) | NOT | JSON with all data about member |
| StartDate | Metadata |  | DATETIME2 | NOT | The time point from which the row is active |
| EndDate | Metadata |  | DATETIME2 |  | The point in time until which the row is active |
| Current | Metadata |  | BIT | NOT | Mark, that the row is active |
| LineageKey | Metadata |  | INT | NOT | Foreign key to audit table |

Table 6 - Product dimension

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| COLUMN NAME | COLUMN TYPE | SCD TYPE | DATA TYPE | NULL | Description |
| ProductKey | Surrogate key |  | INT | NOT | Primary Key in DWH |
| ProductAlterKey | Business key |  | INT | NOT | Primary key in Northwind (упрощённая) |
| Product | Hierarchy | 1 | NVARCHAR(50) | NOT | Product name |
| Category | Hierarchy | 1 | NVARCHAR(50) | NOT | Category name |
| AllAttributes | Metadata | 1 | NVARCHAR(MAX) | NOT | JSON with all data about member |
| LineageKey | Metadata |  | INT | NOT | Foreign key to audit table |

### Facts tables definition

Table 7 - Order facts

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| COLUMN NAME | COLUMN TYPE | DATA TYPE | NULL | Description |
| OrderKey | Degenerate dimension | INT | NOT | Order ID in Northwind (упрощённая) |
| ProductKey | Dimension key | INT | NOT | Foreign key to product table |
| CustomerKey | Dimension key | INT |  | Foreign key to customer table |
| EmployeeKey | Dimension key | INT |  | Foreign key to employee table |
| OrderDateKey | Dimension key | DATE | NOT | Order creation date |
| RequiredDateKey | Dimension key | DATE |  | Оrder fulfillment deadline |
| ShippedDateKey | Dimension key | DATE |  | Order dispatch date |
| UnitPrice | Measure | MONEY |  | Price for one item |
| Quantity | Measure | INT |  | Quantity of items in order |
| Discount | Measure | MONEY |  | Discount amount in dollars |
| SalesAmount | Measure | MONEY |  | UnitPrice \* Quantity |
| SalesAmountWithDiscount | Measure | MONEY |  | SalesAmount - Discount |
| LineageKey | Metadata | INT |  | Foreign key to audit table |

## Designing Data Warehouse Physical Implementation

### Designing File Storage

|  |  |  |
| --- | --- | --- |
| Logical Drive | Filegroups | Notes |
| Disk1 | 1. PRIMARY 2. Default\_FG | * The system tables are in a dedicated filegroup to minimize fragmentation of data files. * The default filegroup is initialized with a single file of 100MB, which should be large enough for the tables it will contain for the near future. |
| Disk2 | Dimention\_Data\_FG | * Putting different tables used in the same join queries in different filegroups. This step will improve performance, because of parallel disk I/O searching for joined data. |
| Disk3 | Dimention\_Index\_FG | * Using different filegroups for heavily accessed tables and the nonclustered indexes will improve performance, because of parallel I/O if the files are located on different physical disks. |
| Disk4 | Order\_Unkown\_Member\_Data\_FG | * This table is likely to grow extremely large, and so it is assigned to a dedicated logical drive to avoid competing for space. |
| Disk5[[2]](#footnote-3) | Order\_1996\_Data\_FG | * As above |
| Disk6 | Order\_1997\_Data\_FG | * As above |
| Disk7 | Order\_Unkown\_Member\_Index\_FG | * Using different filegroups for heavily accessed tables and the nonclustered indexes will improve performance, because of parallel I/O if the files are located on different physical disks. |
| Disk8 | Order\_1996\_Index\_FG | * As above |
| Disk9 | Order\_1997\_Index\_FG | * As above |
| Disk10 | TempDB data files | * TempDB is given its own logical disk to minimize the effect of fragmentation on the data warehouse while enabling fast I/O to a suitable pre-sized TempDB. |
| Disk11 | Database log files | * The logfiles for TempDB, the Staging database, the NorthwindLogs database and the data warehouse are stored on this drive to isolate log activity from data I/O. |
| Disk12 | Backup volume | * This logical drive will be used for backup file storage |

### Partitioning

The **Order** table are partitioned on the **OrderDateKey** column. This column was chosen because most analytics use this field.

Orders for the current month are partitioned into daily partitions, as this is to reflect the ETL load frequency. Orders from previous month are merged to form a single partition per month and Orders from previous years are merged to form a single partition per year, as this reduces management complexity.

### Indexes

The dimension tables each have a clustered index on the surrogate key and a non-clustered index on the alternate key, which includes SCD metadata columns where they exist. Additional non-clustered indexes are created on attribute columns basing on [dimensional models](#_Designing_Dimension_Models_1) and Database Engine Tuning Advisor.

The fact table has clustered columnstore index to maximize aggregations performance. Additional nonclustered indexes are created on dimension key columns to improve performance for joins.

### Compression

All row-stored indexes are compressed using PAGE.

Column-stored indexes compressed using COLUMNSTORE for current and previous year partitions and COLUMNSTORE\_ARCHIVE for archived data.

### Views

Each table has a matching view in the **Reports** schema. These views

* have user-friendly names
* use the NOLOCK query hint to minimize locking
* implement RLS in DWH (is nessesary)

# Designing ETL Solution

## ETL Architecture

Staging

Data Warehouse

Landing zone

Data Sources

Data Cleansing

Master Data Management

Northwind

(упрощённая)



xmlcalendar.ru

Landing



DQS\_STAGING\_DATA



DQS



MDS



DWH

Facts

Dimensions



File system

bcp

bcp



MDS

## Planning Data Extraction

### Data source profiling

Table 8. Data sources.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | Type | Description | Profiling | Auth type | Account |
| Northwind (упрощённая) | SQL Server | Transactional data about customers, employees, products, orders | [Northwind Data Profile.xml](https://dev.azure.com/zinykov/20767/_git/Northwind_BI_Solution?path=/Docs/Northwind%20Data%20Profile.xml)[[3]](#footnote-4) | SQL login | Зинуков Денис Витальевич |
| [xmlcalendar.ru](http://xmlcalendar.ru/) | XML | Information about holidays in Russia | [xmlcalendar/data (github.com)](https://github.com/xmlcalendar/data) | Anonymous |  |

### Identifying New and Modified Rows

Northwind (упрощённая) do not have Data modification time fields, so hash values must be used to look for changes.

Hash values must be calculated using columns that used in Data Warehouse with SHA2\_512 algorithm.

Table 9. Landing.Customers.

|  |  |
| --- | --- |
| Column | Used in Hash |
| CustomerID |  |
| CompanyName | + |
| ContactName | + |
| ContactTitle | + |
| City | + |
| Country | + |
| Phone | + |
| Fax | + |

Table 10. Landing.Products.

|  |  |
| --- | --- |
| Column | Used in Hash |
| ProductID |  |
| ProductName | + |
| SupplierID |  |
| CategoryID | + |
| UnitPrice |  |

Table 11. Landing.Categories.

|  |  |
| --- | --- |
| Column | Used in Hash |
| CategoryID |  |
| CategoryName | + |
| Description |  |

Table 12. Landing.Employees.

|  |  |
| --- | --- |
| Column | Used in Hash |
| EmployeeID |  |
| LastName | + |
| FirstName | + |
| Title | + |
| TitleOfCourtesy | + |
| City | + |
| Country | + |

Table 13. Landing.Orders.

|  |  |
| --- | --- |
| Column | Used in Hash |
| OrderID |  |
| CustomerID | + |
| EmployeeID | + |
| OrderDate | + |
| RequiredDate | + |
| ShippedDate | + |
| ShipCity |  |
| ShipCountry |  |

Table 14. Landing.Order Details.

|  |  |
| --- | --- |
| Column | Used in Hash |
| OrderID |  |
| ProductID |  |
| UnitPrice | + |
| Quantity | + |
| Discount | + |

### Planning Extraction Windows

Table 15. Planning extraction.

|  |  |  |
| --- | --- | --- |
|  | Northwind (Упрощённая) | xmlcalendar.ru |
| How frequently is new data generated in the source systems, and for how long is it retained? | Every day, permanently | Once or twice a year, permanently |
| What is the acceptable delay between changes to source systems and reports? | One day | One day |
| How long does data extraction take? | Less 1 minute | Less 1 minute |
| During what time periods are source systems least heavily used? | Workdays from 09:00:00 to 17:00:00 | No |

Table 16. Extraction Windows.

|  |  |
| --- | --- |
| Data source | Extraction windows |
| Northwind (Упрощённая) | Workdays from 17:00:00 to 09:00:00 and weekend all day |
| xmlcalendar.ru | All day |

## Planning Data Transformation

### Data Flow Diagrams

#### Dimensions

Figure 2 - Schema of SSIS package that load dimension.

Landing



DQS\_STAGING\_DATA



DQS



MDS



DWH

**Filter modified rows**

1. calculate hash in SP
2. Extract data calling view in SSIS OLE DB Source

**Data cleaning** with SSIS DQS Cleansing Transformation

Load with SSIS OLE DB Destination

SSIS Multicast Transformation

**Add MDS Columns and change datatypes** with SSIS Derived Column Transformation

Load into MDS staging tables with SSIS OLE DB Destination

**Validate data with MDS** with SSIS execute SQL tasks

**Lookup AllAttributes columns** from DQS with SSIS Lookup Transformation

**Add metadata columns** with SSIS Derived Column Transformation

**Filter on LastChgDateTime and succeeded validation and calculating AllAttributes column** with T-SQL direct entry in SSIS OLE DB Source

**Lookup existing rows** from DWH with SSIS Lookup Transformation

**Insert new rows (generate surrogate key)** with SSIS OLE DB Destination

**Start audit** with SSIS execute SQL task

**End audit** with SSIS execute SQL task

**Update SCD1 rows** with SSIS OLE DB Command Transformation

Figure 3 - Load of Customer dimension

Northwind (упрощённая)

Customers

Customer

Customer, ContactName, ContactTitle, Phone, Fax

Validate data with MDS

Extract data with BCP

Start audit

Filter modified rows

Filter on LastChgDateTime and succeeded validation

Add metadata columns

Lookup existing rows

Insert new rows (generate surrogate key)

Update SCD1 rows

Update and Insert SCD2 rows (generate surrogate key)

End audit

Data cleaning with DQS

Figure 4 - Load of Date dimension

xmlcalendar

.\ru

Date

Validate data with MDS

Extract data from XML

Start audit

Filter modified rows

Get start and load dates from Landing.Orders

Add unknown member

Populate Date dimension from 01-01 first year to 12-31 last +1 year

End audit

Data cleaning with DQS

Figure 5 - Load of Employee dimension

Northwind (упрощённая)

Employees

Employee

Employee, TitleOfCourtesy

Validate data with MDS

Extract data with BCP

Start audit

Filter modified rows

Filter on LastChgDateTime and succeeded validation

Add metadata columns

Lookup existing rows

Insert new rows (generate surrogate key)

Update SCD1 rows

Update and Insert SCD2 rows (generate surrogate key)

End audit

Data cleaning with DQS

Figure 6 - Load of Product dimension

Northwind (упрощённая)

Products

Product

Product, Category

Validate data with MDS

Filter on LastChgDateTime and succeeded validation

Add metadata columns

Lookup existing rows

Insert new rows (generate surrogate key)

Update SCD1 rows

End audit

Categories

Extract data with BCP

Start audit

Filter modified rows

Data cleaning with DQS

#### Facts

Figure 7 - Schema of SSIS package that load facts.

Landing



DWH

**Create Hash** with calling SP from SSIS execute SQL task

1. **Create filegroups**
2. **Update Partition Schema**
3. **Create load table**

with SSIS execute SQL task

**Start audit** with SSIS execute SQL task

**Extract data using hash** with view in SSIS OLE DB source

**Lookup Dimension Key** with SSIS Lookup transformations

**Calculate measures & Replace NULLs in dimensions keys with unknown dimensions members** with SSIS Derived Column Transformation

**Load Stage Table** with SSIS OLE DB Destination

1. **Create Load Table Constraints**
2. **Switch Partition**

with SSIS execute SQL task

1. **Optimize Partitions Monthly**
2. **Optimize Partitions Yearly**
3. **End audit**

with SSIS execute SQL task

**Landing update hash table** with SSIS execute SQL task

Figure 8 - Load of Order facts

Northwind (упрощённая)

Orders

Update Partition Schema

Create load table

Start audit

Order

End audit

Order Details

Create filegroups

Set start load date

Switch partition

Insert or update data for previous dates

Drop load table

Extract data with bcp

Extract data with bcp

Create Hash

Extract data using hash

Lookup Customer Key

Lookup Employee Key

Lookup Product Key

Customer

Employee

Product

Calculate Sales Amount & Sales Amount with Discount

Replace NULLs in dimensions keys with unknown dimensions members

Optimize partitions Monthly

Optimize partitions Yearly

Load into Landing

### Source To Target Mapping

[Source to target mapping.xlsx](https://dev.azure.com/zinykov/20767/_git/Northwind_BI_Solution?path=/Docs/Source%20to%20target%20mapping.xlsx&version=GBmaster&_a=contents)

### Handling Errors

Event handling OnError for SSIS packages that load DWH tables

Get parameter values

***Task type:*** *Expression Task*

**Expression:** User::ParametersValues = Concatenate parameters and variables in way “Name = Value”

Error notification

***Task type****: Execute Package Task*

**ReferenceType:** Project Reference

**PackageNameFromProjectReference**: Error notification.dtsx

**Parameter Bindings**

|  |  |
| --- | --- |
| Child package parameter | Binding parameter or variable |
| ErrorCode | System::ErrorCode |
| ErrorDescription | System::ErrorDescription |
| ExecutionInstanceGUID | System::ExecutionInstanceGUID |
| FailedConfigurations | System::FailedConfigurations |
| LineageKey | User::LineageKey |
| MachineName | System::MachineName |
| PackageName | System::PackageName |
| ParametersValues | User::ParametersValues |
| SourceID | System::SourceID |
| SourceName | System::SourceName |
| StartTime | System::StartTime |
| UserName | System::UserName |

### Integration Services (SSIS) Logging

Used in all SSIS packages for all containers.

Events types with all columns

1. OnError
2. DiagnosticEx

Table 17 - Provider types

|  |  |
| --- | --- |
| Name | Configuration |
| SSIS log provider for SQL Server | Logs DB |
| SSIS log provider for Text files | TextLog |

# Designing BI Delivery Solution

Power BI usage scenarios: [On-premises reporting](https://learn.microsoft.com/en-us/power-bi/guidance/powerbi-implementation-planning-usage-scenario-on-premises-reporting)

## Reporting Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| Requirement | Audience | Tool | Notes |
| See sales amount with discount by customers and products | Sales VP | Power BI |  |
| Compare sales amount with discount with previous year/quarter/month | CEO, Sales VP | Power BI | [Standard time-related calculations](https://www.daxpatterns.com/standard-time-related-calculations/)  Additionally implement calculated group with context of current period and construction to see previous time periods in context of selected period. |
| See discount by employees and their geography | Sales Representative | Power BI |  |
| See new and returned customers | Sales VP | Power BI | [New and returning customers](https://www.daxpatterns.com/new-and-returning-customers/)   * Absolute.   A customer is considered new the first time they buy a product, regardless of any filter present in the report. |
| Cluster customers based on spending volume | Sales VP | Power BI | [Dynamic segmentation](https://www.daxpatterns.com/dynamic-segmentation/)   * Basic pattern.   2 variants based on Sales amount and Sales amount with Discount. |
| Product ABC classification | Sales VP | Power BI | [ABC classification](https://www.daxpatterns.com/abc-classification/)   * Dynamic ABC classification. * Finding the ABC class. |
| Product basket analysis | Inside Sales Coordinator | Power BI | [Basket analysis](https://www.daxpatterns.com/basket-analysis/)   * Optimized pattern |
| See number of orders, sales amount, sales amount with discount in progress | Inside Sales Coordinator | Power BI | [Events in progress](https://www.daxpatterns.com/events-in-progress/)   * Open orders with snapshot |
| Receive email notifications about errors in the ETL process | Database administrators team | Paginated report |  |
| Notification of errors and raw values in data cleansing processes | Data stewards | Paginated report |  |

# Development operations

The development workflow is described in article under the link [Version control workflow](https://learn.microsoft.com/en-us/azure/devops/repos/git/gitworkflow?view=azure-devops).

## Solution artifacts

|  |  |  |  |
| --- | --- | --- | --- |
| Solution Folder | Project name | Project type | Description |
| Data Engineering | DQS\_STAGING\_DATA | Database | Staging database. Stores data after DQS cleansing. |
| Data Engineering | NorthwindDW | Database | Data Warehouse database. |
| Data Engineering | NorthwindETL | SSIS | SQL Server Integration Services project provides ETL |
| Data Engineering | NorthwindLanding | Database | Landing database. Stores data from data sources as is. |
| Data Engineering | NorthwindLogs | Database | Database used for storing SSIS logs and Data Warehouse metadata. |
| Data Analysis | Monitoring | SSRS | Paginated reports with monitoring |
| Data Analysis | SalesReports | SSRS | Paginated reports with sales information |
| Tests | NorthwindETLTest | Unit Test | Implements Functional ETL test |

## Solution environments

|  |  |
| --- | --- |
| Debug | for debugging changes in developer’s local machine |
| Test | for automated building test assembly and functional testing solution |
| Release | for automated building release assembly and release changes |

## Configuring debug environment

For automating debugging process, it is necessary:

1. Replace in solution files with Find and Replace tool:
   1. computer name
   2. repository folder path
2. To set next configurations in Visual Studio
   1. In Solution properties open “Configure Startup Projects”
      1. Tick “Single startup project” and chose “NorthwindETL”
   2. In database projects properties open “Debug” page
      1. set Target Connection String with you localdb instance
      2. tick in Deployment Options:
         1. Deploy database properties
         2. Always re-create database (before first release)
         3. Block incremental deployment if data loss might occur
   3. In SSIS projects properties open “Debugging”
      1. Set Start Action
         1. StartAction ExecutePackage
         2. StartObjectID Transform and load.dtsx
3. In File Explorer open repository folder
   1. Create folders “logs”
   2. Create folder “NoChange” and inside files
      1. Customer.csv
      2. Employee.csv
      3. Product.csv
   3. In folder “IngestData” create folder “TestData”
   4. Copy folder “OriginalData” into “TestData” and rename it into “1997-12-31”

## Unit tests

For debugging stored procedures, it is recommended to create SQL Server unit test projects.

More information under the link [Verifying Database Code by Using SQL Server Unit Tests](https://learn.microsoft.com/en-us/sql/ssdt/verifying-database-code-by-using-sql-server-unit-tests).

## Functional ETL test

The NorthwindETLTest project is designed for functional testing of ETL.

The project consists of 2 classes:

|  |  |
| --- | --- |
| Class name | Description |
| NorthwindDataTest | implements validation of values in data warehouse tables after loading |
| NorthwindTest | simulates the work of ETL for several days |

Configure tests:

1. Update parameters in test.runsettings

|  |  |
| --- | --- |
| Parameter name | Description |
| LoadDateInitialEnd | Date in format “YYYY-MM-DD hh:mm:ss” with last initial load date for ETL. For correct work it must be synchronized with partitions in DWH. |
| LoadDateIncrementalEnd | Date in format “YYYY-MM-DD hh:mm:ss” with last incremental load date for ETL. You can chose number of iterations in test. |
| DBFilesPath | Part of the path to the folder containing the database files. Corresponds to the properties of the SQL Server instance. |
| SQLServerFiles | Part of the path to the folder containing the database files. Corresponds to the properties of the SQL Server instance. |
| BuildConfiguration | Visual Studio Solution configuration. It is recommended to use value “Debug” for launching in Test Explorer. |
| ExternalFilesPath | Local copy of repository folder path |
| XMLCalendarFolder | Local copy of XMLCalendar repository folder path |

1. Tick file test.runsettings in Configure Run Settings

Run tests:

1. Deploy Solution
2. Deploy SSIS project
3. Open Test Explorer and run “NorthwindTest”

Test trace is published in TestResults folder under repository folder.

There are Xevents and SSIS text logs in logs under repository folder.

## Continuous integration/continuous deployment

[CI/CD baseline architecture with Azure Pipelines](https://learn.microsoft.com/en-us/azure/devops/pipelines/architectures/devops-pipelines-baseline-architecture)

**PR pipeline**

Build

Deploy on test

Functional ETL test



master

branch

Develop

branch

**Azure repos**

release

**PR pipeline**

Build

Deploy on test

Functional ETL test



**CI pipeline**

Build



**CD pipeline**

Deploy on production



# Operation of BI system

## ETL Operations

### Execution context

It is recommended to create environment in SSIS catalog and map NorthwindETL project parameters and environment variables.

Table 18. Environment variables

| Environment variables | Data Type | Description |
| --- | --- | --- |
| DBFilesPath | String | Default path for new database files which are created in fact load packages. |
| DQS\_STAGING\_DATA\_DatabaseName | String | Name of staging database where writing data after DQS cleansing. |
| DQS\_STAGING\_DATA\_ServerName | String | Name of server with staging database where writing data after DQS cleansing. |
| DQSServerName | String | Name of Data Quality Server |
| DWHDatabaseName | String | Name of Data Warehouse database. |
| DWHServerName | String | Name of Data Warehouse server. |
| ExternalFilesPath | String | A file system location for saving:   1. data from data sources that don’t use SQL table as destination 2. logs files (SSIS log provider for Text files & Xevents trace files) 3. CSV staging files 4. SQL scripts (needed for CI/CD) 5. test results |
| LandingDatabaseName | String | Name of database used for raw data from data sources. |
| LandingServerName | String | Name of server used for raw data from data sources. |
| LogsDatabaseName | String | Name of database used for storing SSIS logs and Data Warehouse metadata |
| LogsServerName | String | Name of server with database used for storing SSIS logs and Data Warehouse metadata |
| MDSDatabaseName | String | Name of database used for Master data services catalog. |
| MDSServerName | String | Name of server with database used for Master data services. |
| XMLCalendarFolder | String | Path to local repository of xmlcalendar project. |
| CutoffTime | DateTime | The point in time up to which changes are loaded from data sources |
| LoadDateInitialEnd | DateTime | The point in time up to which changes are loaded from data sources on first load |

### Execution account

It is recommended that you create a **dedicated domain account** to run SSIS packages in a production environment and map it to the **SQL Server Agent Proxy** account.

## Data Warehouse Operations

### Reorganizing or rebuilding indexes

Reorganizing or rebuilding indexes uses for partitions in read-write filegroups. Implemented in stored procedure [OptimizeIndexes.sql](https://dev.azure.com/zinykov/20767/_git/Northwind_BI_Solution?path=/NorthwindDW/Maintenance/Store%20Procedures/OptimizeIndexes.sql) with the following logic.

IF avg\_fragmentation\_in\_percent between 5 and 30 THEN reorganize

IF avg\_fragmentation\_in\_percent > 30 THEN rebuild

Implementation described in paragraph “Maintenance subplan: DWH Maintenance”.

### Updating data distribution statistics

Uses the **sp\_updatestats** stored procedure.

Implementation described in paragraph “Maintenance subplan: DWH Maintenance”.

### Managing partitioned tables

The manage partitions for each fact table occurs in the fact load package.

Steps:

1. Every day:
   1. Create filegroup if does not exist.
   2. Create partition for new portion of data.
2. Every first night from Friday to Saturday in month merges day partitions for the previous month in a month partition.
3. Every first night from Friday to Saturday in year merges month partitions for the previous year in a year partition.

### Security

|  |  |
| --- | --- |
| Role | Grants |
| **dwh\_user** | 1. SELECT ON SCHEMA::[Reports] 2. EXECUTE ON SCHEMA::[Reports] 3. VIEW DEFINITION |

## DQS Operations

### Configuration

* **General Settings**: Specify the threshold values for data cleansing and data matching, and whether to enable notifications for profiling in Data Quality Client. These threshold values are used by DQS during the computer-assisted cleansing and matching activities in a data quality project.
* **Log Settings**: The log files in DQS record the activities performed in DQS and are useful for tracking operational issues during maintenance and troubleshooting. You can filter the messages that you want to be logged for various DQS features (domain management, knowledge discovery, cleansing, matching, and reference data services) and DQS modules based on the severity level of the events.

### Security

Roles:

* The **DQS Administrator** (dqs\_administrator role) can do everything in the scope of the product. The administrator can edit and execute a project, create and edit a knowledge base, terminate an activity, stop a process within an activity, and can change the configuration and Reference Data Services settings. The DQS Administrator cannot, however, install the server or add new users. The database administrator must do that.
* The **DQS KB Editor** (dqs\_kb\_editor role) can perform all of the DQS activities, except for administration. The KB Editor can edit and execute a project, and create and edit a knowledge base. They can see the activity monitoring data, but cannot terminate or stop an activity or perform administrative duties.
* The **DQS KB Operator** (dqs\_kb\_operator role) can edit and execute a project. They cannot perform any kind of knowledge management; they cannot create or change a knowledge base. They can see the activity monitoring data, but cannot terminate an activity or perform administrative duties.

The database administrator (DBA) creates DQS users and associates them with DQS roles in SQL Server Management Studio. The DBA manages their permissions by adding SQL Logins as users of the DQS\_MAIN database and associating each user with one of the DQS roles. Each role is granted permissions to a set of stored procedures on the DQS\_MAIN database. The three DQS roles are not available for the DQS\_PROJECTS and DQS\_STAGING\_DATA databases.

## MDS Operations

### Security

Types of Users:

* Those who access data in the Explorer functional area.
* Those who have the ability to perform administrative tasks in areas other than Explorer.

[Security - SQL Server Master Data Services](https://learn.microsoft.com/en-us/sql/master-data-services/security-master-data-services?view=sql-server-ver15)

Additionally, in MDS database created roles:

* The VSTS\_AgentService\_G39071 role can select and execute in schema stg. This role is necessary for ETL.
* The RDLexec role can select from master data views and schema stg. This role is necessary for monitoring reports in Power BI Report Server.

### Cleaning up transactions and staging tables

Cleaning up transactions and staging tables implemented with stored procedures in Data Warehouse and automated with DWH maintenance package. Transactions and data in staging tables older than 7 days are cleared by default.

## Reporting Services Operations

### Managing Scheduled Tasks

Considerations for schedules:

* Schedules depend on the SQL Server Agent. If SQL Server Agent is not running, scheduled tasks will not run.
* Using shared schedules enables you to centrally pause, resume, and modify multiple scheduled tasks in a specific location. Using object-specific schedules requires managing each schedule individually.
* A report server uses the time zone of the computer that hosts it.
* If you change the time zone of a server that hosts Power BI Report Server, you must restart the Reporting Services service for the time zone change to take effect. When you change the time zone of a report server, existing schedules retain the same times in the new time zone. For example, a task that was scheduled to run at 2:00 in the old time zone will be scheduled to run at 2:00 in the new time zone.

## Backing up BI Solution

**RTO**: 1 hour

**RPO**: 30 minutes

To ensure RTO and RPO has been created maintenance plan which is described in paragraph “Maintenance plan: NorthwindBI”.

Recommendations for backing up BI Solution resources after installation and configuration:

1. SSIS Master key
2. Generate scripts to recreate:
   1. The ##MS\_SSISServerCleanupJobLogin## login.
   2. The dbo.sp\_ssis\_startup stored procedure.
   3. The SSIS Server Maintenance Job SQL Server Agent job.
   4. The ##MS\_dqs\_db\_owner\_login## login
   5. The ##MS\_dqs\_service\_login## login
   6. The DQInitDQS\_MAIN stored procedure in the master database
3. Report Server Encryption key
4. Report Server Configuration files
   1. Rsreportserver.config
   2. Rssvrpolicy.config
   3. Web.config

To ensure RTO and RPO for DWH database implemented partial backup strategy.

Steps of partial backup strategy:

1. Changes compression to COLUMNSTORE\_ARCHIVE for partitions with fact data and nonclustered indexes of the year before last.
2. Sets filegroups with fact data and nonclustered indexes of the year before last READONLY.
3. Every night creates READONLY filegroups backup if it was not done.
4. Every night from Saturday to Sunday creates full backup for READ\_WRITE\_FILEGROUPS.
5. Every night but from Saturday to Sunday creates differential backup for READ\_WRITE\_FILEGROUPS.

Implementation described in paragraph “Maintenance subplan: DWH Maintenance”.

## Manual Operational Tasks

### Managing database files

Data Warehouse will create new filegroups and files for fact data in default file system folder (parameter DBFilesPath).

**DBA must** add information about database fact table files **annually**.

To point location for new database files use stored procedure [NorthwindDW].[Maintenance].[InsertDatabaseFilesData].

Table 19 - [Maintenance].[InsertDatabaseFilesData] parameters.

|  |  |  |
| --- | --- | --- |
| Parameter | Data type | Description |
| @FactTableName | NVARCHAR(100) | The name of the fact table which data will be stored in the file. |
| @FilePath | VARCHAR(500) | Path to folder for database file |
| @IsClustered | BIT | If 0 then file for filegroup for non-clustered indexes  If 1 then file for filegroup for clustered indexes |
| @Year | INT | The period of time for which data will be stored in the file |

The information about partitions, filegroups and files places in tables [NorthwindLogs].[Integration].[DatabaseFiles] and [NorthwindDW].[Maintenance].[DatabaseFiles].

## Automating Operational Tasks

### SQL agent jobs

Job NorthwindBI.ExtractNorthwind(упрощённая)

Category Northwind BI

Schedule 01:00:00 AM every day

Steps:

|  |  |
| --- | --- |
| Step | Extract Northwind (упрощённая) |
| Type | SQL Server Integration Services Package |
| Package source | SSIS catalog |
| Package | \SSISDB\NorthwindBI\NorthwindETL\Extract Northwind (упрощённая).dtsx |
| Environment | Release |

Job NorthwindBI.ExtractXmlcalendar

Category Northwind BI

Schedule 01:00:00 AM every day

Steps:

|  |  |
| --- | --- |
| Step | Extract Xmlcalendar |
| Type | SQL Server Integration Services Package |
| Package source | SSIS catalog |
| Package | \SSISDB\NorthwindBI\NorthwindETL\Extract Xmlcalendar.dtsx |
| Environment | Release |

Job NorthwindBI.TransformAndLoad

Category Northwind BI

Schedule 02:00:00 AM every day

Steps:

|  |  |  |
| --- | --- | --- |
| Step | Transform and load | |
| Type | SQL Server Integration Services Package | |
| Package source | SSIS catalog | |
| Package | \SSISDB\NorthwindBI\NorthwindETL\Transform and load.dtsx | |
| Environment | Release | |
| Step | **DWH Maintenance** | |
| Type | SQL Server Integration Services Package | |
| Package source | SQL Server | |
| Package | \Maintenance Plans\NorthwindBI | |
| Set values | Property Path | Value |
| \Package\DWH Maintenance.Disable | false |
| Step | **Copy DatabaseFiles info** | |
| Type | SQL Server Integration Services Package | |
| Package source | SSIS catalog | |
| Package | \SSISDB\NorthwindBI\NorthwindETL\Maintenance copy DatabaseFiles.dtsx | |
| Environment | Release | |

### Maintenance plan: NorthwindBI

#### Maintenance subplan: DWH Maintenance

**Schedule**: described in paragraph SQL agent jobs

**Runs as:** SQL Server Agent Proxy

Check DWH integrity

***Task type****: Check Database Integrity Task*

**Databases**

1. NorthwindDW

Include indexes

Physical Only

Optimize Indexes

***Task type****: Execute T-SQL statement Task*

**Script:**

USE [NorthwindDW];

GO

EXECUTE [Maintenance].[OptimizeIndexes];

GO

Update statistics

***Task type****: Update Statistics Task*

**Databases**

1. NorthwindDW

**Update**: All existing statistics

**Scan type**: Full scan

Backup DWH

***Task type****: Execute T-SQL statement Task*

Script text: [DWH Backup.sql](https://dev.azure.com/zinykov/NorthwindBI/_git/Northwind_BI_Solution?path=%2FScripts%2FDWH%20Backup.sql&version=GBmaster&_a=contents)

#### Maintenance subplan: Full backup

**Schedule**: Occurs every week on Sunday at 1:00:00.

**Runs as:** SQL Server Agent Proxy

Check DWH integrity

***Task type****: Check Database Integrity Task*

**Databases**

1. DQS\_MAIN
2. DQS\_PROJECTS
3. DQS\_STAGING\_DATA
4. NorthwindLogs
5. master
6. MDS
7. msdb
8. ReportServer
9. ReportServerTempDB
10. SSISDB

Include indexes

Physical Only

Backup DWH

***Task type:*** *Backup database task*

**Type:** Full

**Databases:**

1. DQS\_MAIN
2. DQS\_PROJECTS
3. DQS\_STAGING\_DATA
4. NorthwindLogs
5. master
6. MDS
7. msdb
8. ReportServer
9. SSISDB

**Backup to:** disk

**Backup set will expire** after 30 days

**Set backup compression:** Compress backup

#### Maintenance subplan: Differential backup

**Schedule:** Occurs every day every 12 hour(s) between 0:00:00 and 23:59:59.

**Runs as:** SQL Server Agent Proxy

Check DWH integrity

***Task type****: Check Database Integrity Task*

**Databases**

1. DQS\_MAIN
2. DQS\_PROJECTS
3. DQS\_STAGING\_DATA
4. NorthwindLogs
5. master
6. MDS
7. msdb
8. ReportServer
9. ReportServerTempDB
10. SSISDB

Include indexes

Physical Only

Backup DWH

***Task type:*** *Backup database task*

**Type:** Differential

**Databases:**

1. DQS\_MAIN
2. DQS\_PROJECTS
3. DQS\_STAGING\_DATA
4. NorthwindLogs
5. master
6. MDS
7. msdb
8. ReportServer
9. SSISDB

**Backup to:** disk

**Backup set will expire** after 14 days

**Set backup compression:** Compress backup

#### Maintenance subplan: Log backup

**Schedule:** Occurs every day every 30 minute(s) between 0:00:00 and 23:59:59.

**Runs as:** SQL Server Agent Proxy

Backup DWH

***Task type:*** *Backup database task*

**Type:** Transaction log

**Databases:**

1. MDS
2. ReportServer
3. SSISDB

**Backup to:** disk

**Backup set will expire** after 1 days

**Set backup compression:** Compress backup

### ETL execution duration

At weekends nights ETL will be longer because of maintenance operations:

* Every night from Saturday to Sunday creates full database backup
* Every first night in month from Friday to Saturday merges daily partitions in monthly. Additionally, every January merges monthly partitions in yearly.

## Disaster recovery plan

1. Installation
   1. Install SQL server 2022
   2. Install DQS Server
   3. Install MDS
   4. Install Power BI Report Server
2. Restore DQS using [Backing Up and Restoring DQS Databases](https://learn.microsoft.com/en-us/sql/data-quality-services/backing-up-and-restoring-dqs-databases?view=sql-server-ver15#BackupRestore)
3. Restore MDS using [SQL Server Master Data Services (MDS) Database Restore Steps](https://www.mssqltips.com/sqlservertip/4295/sql-server-master-data-services-mds-database-restore-steps/)
4. Restore SSIS using [Moving the SSISDB Catalog on a new SQL Server instance](https://www.sqlshack.com/moving-the-ssisdb-catalog-on-a-new-sql-server-instance/)
5. Restore Report Server:
   1. Restore ReportServer database
   2. Create ReportServerTempDB database
   3. Run the CatalogTempDB.sql script in the \Program Files\Microsoft Power BI Report Server\PBIRS\ReportServer folder
   4. Restore Report server Encryption Keys using [Restore encryption keys - Report Server Configuration Manager](https://learn.microsoft.com/en-us/sql/reporting-services/install-windows/ssrs-encryption-keys-back-up-and-restore-encryption-keys?view=sql-server-ver15#bkmk_restore_configuration_manager)
   5. Restore Report server configuration files in the \Program Files\Microsoft Power BI Report Server\PBIRS\ReportServer folder
6. Restore NorthwindLogs database
7. Restore Landing database
8. Restore DQS\_STAGING\_DATA database
9. Restore msdb database
10. Restore Data Warehouse with script [RestoreDatabase.sql](https://dev.azure.com/zinykov/20767/_git/Northwind_BI_Solution?path=/NorthwindDW/Scripts/RestoreDatabase.sql)
11. Create users with script [CreateUsers.sql](https://dev.azure.com/zinykov/20767/_git/Northwind_BI_Solution?path=/NorthwindDW/Scripts/CreateUsers.sql)

# Monitoring and Optimizing BI Solution

## Creating a Performance Baseline

Before applying an effective health monitoring process, it is necessary to have a set of resource utilization metrics with which to compare future measurements. It is necessary to collect baseline resource utilization metrics for each server or service in the BI solution and collect the data over a sufficient period to reflect normal workload cycles. These workload cycles should include:

* Data warehouse load operations.
* Power BI reports processing operations.
* Normal everyday query activity.
* Peak-time query activity.

## Monitoring ETL

### Package Execution Reports

SSIS catalog reports:

1. **Integration Services Dashboard**. This report provides a central summary that shows details of package executions. For each package execution listed in this report, you can drill into three sub-reports: Overview, All Messages, and Execution Performance.
2. **All Executions.** This report provides details of all package executions on the server, and you can filter it to show executions within a specified date range.
3. **All Connections.** This report shows details of all connections that the package used during executions, including connection strings and whether the connection failed or succeeded.
4. **All Operations.** This report shows details of all operations that occurred on the server, including package deployments, executions, and other administrative operations.
5. **All Validations.** This report shows details of all validations that SSIS performed for packages.

[SSRS project Monitoring](https://dev.azure.com/zinykov/20767/_git/Northwind_BI_Solution?path=/NorthwindPBIRS/Monitoring) reports:

1. **Monitoring**. This report provides information about loading duration, detailed information about Data Warehouse partitions, row validating statuses in DQS and MDS and errors in ETL process.
2. **SSIS log**. This report provides visual representation of sysssislog table.

## Integration.Lineage

Every table in Data Warehouse has foreign key to **[Integration].[Lineage]**. This table is coping every load in Logs DB.

## Monitoring and optimizing Data Warehouse

### Considerations for Data Warehouse Performance

#### Data warehouse workloads:

* **ETL data loads.** These workloads are usually regularly scheduled events that load new or updated data into the tables in the data warehouse.
* **Power BI reports data models processing.** These workloads usually occur after each ETL data load, to refresh Power BI reports data models with the new data.
* **Report queries.** These workloads occur when users create a report or a user-defined data model in Excel that requires the retrieval of data is retrieved from data warehouse tables.
* **Operational activities.** These workloads are usually scheduled operations, such as index maintenance or database backups.

#### SQL Server settings

SQL Server provides various configuration settings that affect the way the service uses hardware resources. In particular, can be used the Maximum Server Memory configuration setting to specify the maximum amount of system memory that SQL Server can use, and can be used the ALTER SERVER CONFIGURATION statement to control how SQL Server uses CPU threads and simulate non-uniform memory access (NUMA).

### Monitoring tools

* [Performance Monitor](https://learn.microsoft.com/en-us/sql/relational-databases/performance-monitor/monitor-resource-usage-system-monitor?view=sql-server-ver15)
* [SQL Server Profiler](https://learn.microsoft.com/en-us/sql/relational-databases/sql-trace/sql-trace?view=sql-server-ver15)
* [Data Collector](https://learn.microsoft.com/en-us/sql/relational-databases/data-collection/data-collection?view=sql-server-ver15)
* [Extended Events](https://learn.microsoft.com/en-us/sql/relational-databases/extended-events/extended-events?view=sql-server-ver15)

### Turning tools

* [Database Engine Tuning Advisor](https://learn.microsoft.com/en-us/sql/relational-databases/performance/database-engine-tuning-advisor?view=sql-server-ver15)
* [Query Store](https://learn.microsoft.com/en-us/sql/relational-databases/performance/monitoring-performance-by-using-the-query-store?view=sql-server-ver15)

### Using Resource Governor to Balance Resource Utilization

Table 20. SQL Server resource pools

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Name | Minimum CPU % | Maximum CPU % | Minimum Memory % | Maximum memory % | Minimum I/O operations per second (IOPS) per disk volume | Maximum I/O operations per second (IOPS) per disk volume |
| Low Priority | 0 | 50 | 0 | 50 | 0 | 65 536 |
| High Priority | 20 | 90 | 20 | 90 | 65 536 | 0 |

Table 21. SQL Server workload groups

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Name | Resource pool name | Importance | Maximum Requests | CPU Time (sec) | Memory Grant % | Memory Grant Time-out (sec) | Degree of Parallelism |
| User Queries | Low Priority | Low | 10 | 50 | 50 | 20 | 1 |
| ETL | High Priority | High | 100 | 80 | 80 | 30 | 4 |

Conditions for dbo.fn\_classify:

IF USER\_NAME IN

AzPipelineAgent

SQLAGENT

RDLexec

THEN Workload group “ETL”

ELSE Workload group “User Queries”

### Troubleshooting Data Warehouse Performance

1. Performance monitor counters ([Performance Monitor Template](https://dev.azure.com/zinykov/NorthwindBI/_git/Northwind_BI_Solution?version=GBmaster&path=/Monitoring/PerformanceMonitoring.xml))[[4]](#footnote-5):
   1. Memory: Available MBytes
   2. Paging File: % Usage
   3. Physical Disk:Avg. Disk sec/Read
   4. Physical Disk: Avg. Disk sec/Write
   5. Physical Disk: Disk Reads/sec
   6. Physical Disk: Disk Writes/sec
   7. Processor: % Processor Time
   8. SQLServer:Buffer Manager: Buffer cache hit ratio
   9. SQLServer:Buffer Manager: Page life expectancy
   10. SQLServer:General Statistics: User connections
   11. SQLServer:Memory Manager: Memory grants pending
   12. SQLServer:SQL Statistics: Batch requests/sec
   13. SQLServer:SQL Statistics: Compilations/sec
   14. SQLServer:SQL Statistics: Recompilations/sec
   15. System: Processor queue length
   16. SQLServer:Resource Pool Stats: CPU control effect %
   17. SQLServer:Workload Group Stats: CPU usage %
   18. SQLServer:Workload Group Stats: Reduced memory grants/sec
2. SQL Server Profiler ([SQL Server Profiler Template](https://dev.azure.com/zinykov/NorthwindBI/_git/Northwind_BI_Solution?version=GBmaster&path=/Monitoring/Monitor%20Data%20Warehouse%20Query%20Activity.tdf)):
   1. events
      1. SQL: BatchCompleted
      2. SQL: StmtCompleted
   2. columns
      1. ApplicationName
      2. DatabaseName
      3. Duration
      4. EndTime
      5. LoginName
      6. Reads
      7. RowCounts
      8. SPID
      9. StartTime
      10. TextData
   3. filters
      1. DatabaseName LIKE %NorthwindDW%

## Monitoring DQS

The **Activity Monitoring** screen in Data Quality Client displays detailed information about each activity performed on a Data Quality Server. Enables the DQS administrators to control an activity or a process within an activity by terminating a running activity or stopping a running process within an activity, if required. The data is displayed for knowledge discovery, domain management, matching policy, cleansing, matching, and SQL Server Integration Services (SSIS)-based cleansing.

## Monitoring MDS

[Tracing - SQL Server Master Data Services](https://learn.microsoft.com/en-us/sql/master-data-services/tracing-master-data-services?view=sql-server-ver15)

## Monitoring and optimizing Power BI Report Server

### Considerations for Reporting Services Performance

#### Reporting Services workloads

* **Data retrieval and processing.** When responding to a request for a report (interactively or by a scheduled subscription, cache refresh, or snapshot), Reporting Services loads the definition of the report and uses its datasets to retrieve the required data. Reporting Services then generates the report in an intermediate format for rendering.
* **Report rendering.** When a user views or exports a report, or Reporting Services will deliver a report as a subscription, Reporting Services will use the appropriate rendering extension to render the report into the required format.

#### Reporting Services settings

* **WorkingSetMinimum.** You can add this setting to the RSReportServer.config file to specify the minimum amount of system memory that Reporting Services must have allocated before it will start to release memory resources. By default, this setting is 60 percent of the memory available on the server. If Reporting Services is within this value, the level of memory pressure is considered low.
* **MemorySafetyMargin.** You can use this value to specify a percentage of WorkingSetMaximum. If Reporting Services exceeds this amount of memory, the level of memory pressure is considered medium, and Reporting Services begins to refuse some requests for memory reduce memory allocations.
* **MemoryThreshold.** You can use this value to specify a percentage of WorkingSetMaximum that is higher than MemorySafetyMargin. If Reporting Services exceeds this amount of memory, the level of memory pressure is considered high, and Reporting Services begins to manage requests for memory aggressively.
* **WorkingSetMaximum.** You can add this setting to the RSReportServer.config file to set the maximum amount of system memory that Reporting Services can use. By default, this setting is not included in the RSReportServer.config file, and Reporting Services can access all memory available on the server.

### Monitoring Tools

* [Power BI Desktop](https://learn.microsoft.com/en-us/power-bi/guidance/monitor-report-performance)
* [Power BI report server logs](https://learn.microsoft.com/en-us/sql/reporting-services/report-server/reporting-services-log-files-and-sources?view=sql-server-ver15)

### Tune

* [Optimization guide for Power BI](https://learn.microsoft.com/en-us/power-bi/guidance/power-bi-optimization)

### Using Caching and Snapshots to Optimize Performance

Consider the following factors:

* Reporting Services stores cached reports in an intermediate format that includes data and layout information. It is still necessary to render the report to the requested format after recovering it from cache.
* Cached datasets and reports are based on specific parameter value combinations. Reporting Services creates a cached copy of the report or dataset for each combination of parameter values requested.
* Can be configured a cached object to expire after a specified interval (in minutes), or at a time specified in a schedule. Schedules can be specific to an individual cached object or shared across multiple objects. When a cached object expires, Reporting Services removes it from the cache and the next request results in a new execution with live data.
* Can be preloaded a cached object by creating a cache refresh schedule, or by scheduling a subscription for a cached report with a NULL delivery extension.
* In the event of a server restart, Reporting Services recreates cached objects.

1. <https://www.kimballgroup.com/data-warehouse-business-intelligence-resources/kimball-techniques/dw-bi-lifecycle-method/> [↑](#footnote-ref-2)
2. For every file group is needed dedicated disk. [↑](#footnote-ref-3)
3. For view xml file is needed [Data Profile Viewer](https://learn.microsoft.com/en-us/sql/integration-services/control-flow/data-profile-viewer?view=sql-server-ver15) [↑](#footnote-ref-4)
4. Additional information about performance counters is under the link [BI performance counters (sqlshack.com)](https://www.sqlshack.com/bi-performance-counters/). [↑](#footnote-ref-5)