**SQL Data Warehouse from Scratch**

Video Link = <https://www.youtube.com/watch?v=9GVqKuTVANE&t=5560s>

**Data Warehouse**

Data Warehouse is a subject oriented, integrated, time variant and non-volatile collection of data in support of management’s decision-making process.

* **Subject oriented** : DWs are always focused on business area such as sales, customer, marketing, finance .
* **Integrated** : Combining data from multiple heterogenous sources into a unified, consistent view.
* **Time Variant :** Data is stored in a way that allows historical analysis over time .
* **Non –** **Volatile :** once data is loaded into the warehouse, it is not frequently changed, updated, or deleted.

Imagine your organization doesn’t have a proper data management system. Now you have system and data analyst has to go to this system and start collecting and extracting the data and then he spend days or weeks transforming raw data into something meaningful. Once they develop it then they share it with stakeholders. And you have another source of data and another data analyst doing the same approach to create a power point presentation. There are lots of issues in this process. It takes lots of time to create reports and when users receive the reports, they might be old. Manual process is always slow and stressful, and the more employees involved in the process the more you open to human errors. And if a stakeholder asks for an integrated report from multiple sources, it will be hard.

If your organization has a data warehouse, you will never need to collect data manually. We are going to have an important component called ETL. It is going to extract data from multiple sources and apply some transformations and load it into a data warehouse. So, the data warehouse is going to be a single source of truth for analyzing and reporting. All our reports are going to be consuming from this single point of truth. We can create multiple reports as well as we can create integrated reports from multiple reports. This is process is very organized and fully automated

**ETL**

Our data exists in a source system and now we need to do is to get our data from source and move it to the target. The first step we must take is to specify which data we have to load from the source. This process is known as extract. We just identifying what data we need and pull it out we don’t change anything. And then take this extracted data and do some manipulations, transformations and we are going to change the shape of those data. After those transformations, we are going to load that data to the target.

Upsert

Truncate & Insert

Streaming Processing

Batch Processing

Pull Extraction

Push Extraction

Drop, Create & Insert

Full Load

Extraction Method

Processing Type

Extraction Techniques

Incremental Extraction

Full Extraction

Extraction Types

Load Methods

Upsert

Append

Incremental Load

Merge

Manual Data Extraction

Overwrite

No Historaization

Historaization

Slowly Changing Dimension

Data Enrichment

Data Integration

Derived Column

Data Aggregation

Data Normalization and Standardization

Business Rules and Logics

Outlier Detection

Data Type Casting

Data Handling Unwanted Space

Data Cleansing

Remove Duplication

Data Filtering

Handling Missing Data

Handling Invalid Values

Web Scraping

CDC

Event Based Streaming

API Calls

File Parsing

Database Querying

**Project Plan with Notion**

**A screenshot of a data warehouse project

AI-generated content may be incorrect.**

**Analyze Requirement**

**Project Requirement**

**Building the Data Warehouse (Data Engineering)**

**Objective**

Develop a modern data warehouse using SQL Server to consolidate sales data, enabling analytical reporting and informed decision-making.

Specifications

* **Data Source:** Import data from two source systems(ERP and CRM) provided as CSV files.
* **Data Quality :** Cleanse and resolve data quality issue prior to analysis
* **Integration:** Combine both sources into a single, user-friendly data model designed for analytical queries.
* **Scope:** Focus on the latest dataset only; historization of data is not required.
* **Documentation:** Provide clear documentation of the data model to support both business stakeholders and analytics teams.

**BI: Analytics & Reporting (Data Analysis)**

**Objective**

Develop SQL-based analytics to deliver insight into;

* **Customer Behavior**
* **Product Performance**
* **Sales Trends**

These insights empower stakeholders with key business metrics, enabling strategic decision -making.

**Design the Data Architecture**

Data architect is like a house architect, they design how your data will flow, integrate and be accessed. Data Architecture has to take the decision to choose between 4 major types,

1. Data Warehouse – Suitable if you have only structured data and your organization want to build solid foundations for reporting and business intelligence.
2. Data Lake – Way more flexible that data warehouse, where you can store not only structured data but as well semi and unstructured data. We use this approach if we have mixed types of data like database table, images, videos and your business want to focus not only on reporting but as well on advanced analytics or machine learning. It is not organized like data warehouse.
3. Data Lakehouse : It is a mix between data warehouse and data lake. We can have flexibility of having different types of data from data lake but still want to structure and organize your data like we do in the warehouse.

There 4 different types of approaches to build data warehouse.

1. Inmon Approach

Sale

EDW

Stage



HR

* In here you have your sources and the first layer you start with the staging where the raw data is landing. And then the next layer you organize your in something called “Enterprise Data Warehouse” where you go and model the data using the 3rdNF. Next layer is “Data Mart” where you goo and take like small subset of the data warehouse and you design it is way that is ready to be consumed from reporting and it only focus only on one topic. After that you go and connect your BI tool to the data marts.

1. Kimball Approach

Sale

Stage

\*\*



HR

* This says building data warehouses is waste of time. What we can do is directly jump from stage layer to data mart. But in the time you might get chaos in the data marts because you are not always focusing in the big picture and you might be repeating same transformations and integrations in different data marts. There is a trade between the speed and consistent data warehouse.



1. Data Vault

Sale

Raw Vault

Business Vault

Stage



HR

* This method says, we need to bring more standards and rules, this divides middle layer into raw vault and business vault. Raw vaults have the original data , business vaults include all the business rules and transformations that prepare the data for the data mart. It is similar to inmon method but it adds more standards and rules to the middle.

1. Medallion Architecture

Gold Layer

Bronze Layer

Silver

Layer



* In here we create 3 layers, bronze, silver and gold layer. Bronze layer is like the stage but having stage layer is very important because having the original data as it is going to help a lot by traceability and finding issues. Then we have silver layer, it is where we perform data transformation and data cleansing, but we do not apply business rules. Gold layer is similar to data mart but there we can build different types of objects not only for reporting but as well for machine learning.

**Design the layers**

1. **Bronze Layer**

* **Definition :** Store raw and unprocessed data as it is from the sources.
* **Objective :** Traceability and Debugging
* **Object Type:** Tables
* **Load Method:** Full Load
* **Data Transformations :**None
* **Data Modeling:** None
* **Target Audience:** Data Engineer

It is important to have the data as it is from the sources because we can go back to the previous layer and investigate the data of specific sources, if something goes wrong.

1. **Silver Layer**

* **Definition:** Clean & Standardized data
* **Objective:** Prepare data for analysis
* **Object Type:** Tables
* **Load Method:** Full Load
* **Data Transformations :** Data Cleaning, Data Standardization, Data Normalization, Derived Columns, Data Enrichment
* **Data Modeling:** None
* **Target Audience:** Data Analysts, Data Engineers

This is the place where we are going to perform basic transformations to prepare the data for the final layer.

1. **Gold Layer**

* **Definition:** Business ready data
* **Objective:** Provide data to be consumed for reporting and analytics.
* **Object Type:** Views
* **Load Method:** None
* **Data Transformation:** Data Integration, Data Aggregation, Business Logic & Rules.
* **Data Modeling:** Star Schema, Aggregated Object, Flat Tables
* **Target Audience:** Data Analysts, Business Users

**Draw the Architecture**

**Define Naming Conventions**

* Set of rules or guidelines for naming anything in the project.

1. Database
2. Schema
3. Tables
4. Stored Procedure

* In this project, all the letters in the word is going to be lower case and separation between words going to be an underscore. Ex: customer\_info
* Use English for all names.
* Avoid reserved words.

**Bronze and Silver Rules**

* All names must start with the source system name, and table names must match their original names without renaming.
* <sourcesystem>\_<entity>
* <sourcessystem>:Name of the source system (e.g, crm, erp)
* <entity> : Exact table name from the source system
* Example: crm\_customer\_info >>> Customer information from the CRM system.

**Gold Rules**

* All names must use meaningful, business-aligned names for tables, starting with the category prefix.
* <category>\_<entity>
* <category>: Describe the role of the table, such as dim (dimensional) or fact (fact table)
* <entity>: Descriptive name of the table, aligned with the business domain(e.g, customer, sales)

**Preparing for the GIthub**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Create Database and Schemas**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Build the Bronze layer**

**Docs & Version**

Data Documenting

Versioning In GIT

**Validation**

Data Completeness and Schema Checks

**Coding**

Data Ingestion

Interview source system expert

**Analyze**

**Analyze Source System**

**Business Context & Ownership Architecture & Technology Stack**

* How is data stored?

(SQL server, Oracle, AWS, Azure)

* What are the integration capabilities?

(API, Kafka, File Extract)

* Who owns the data?
* What Business Process it supports?
* System & Data Documentation
* Data Model & Data Catalog

**Extract & Load**

* Incremental vs Full Loads?
* Data Scope & Historical Needs
* What is the expected size of the extracts?
* Are there any data volume limitation?
* How to avoid impacting the source system’s performance?
* Authentication and Authorization

**‘**

**A screenshot of a computer

AI-generated content may be incorrect.Bronze Layer Table Creation**



* This checks if the table crm\_cust\_info exist under bronze schema.
* Object\_id(‘bronze.crm\_cust\_info’, ‘U’) returns the object id of the table if it exists.
* ‘U’ specifies that the object type being checked is a user table.

**Develop SQL Load Scripts**

A screenshot of a computer

AI-generated content may be incorrect.The method we are going to use to load the data from the source to the data warehouse is  **“Bulk Insert” .** It is a method of loading massive amounts of data very quickly from files like CSV files into a database.

* “TABLOCK” key word of the Bulk Insert statement is used to improve performance by acquiring a table-level lock instead of row level or page level lock when inserting data.

A screen shot of a computer

AI-generated content may be incorrect.

* If we continually execute the bulk insert command , it will insert data again and again making data duplicate. To avoid that kind of scenario, we add truncate command before bulk insert statement.

**Create Stored Procedures**

To get the new content, we need to run above query daily basis. If we have a script of sql that is frequently used, we can create a stored procedure from those scripts.

A screen shot of a computer

AI-generated content may be incorrect.

* Now we need to update this capture errors and bugs in the code.
* We have created a try catch block which will identify errors and handle them.
* Then we need a method to capture ETL duration
* After that we need to calculate the duration of loading bronze layer as “Whole Batch”

**Creating Data Flow**

We are going to draw a simple visual to map the flow of your data. Where it comes from and where it ends. We want to make clear how the data flows through the different layers of our project. This helps us to create data lineage.

**Silver Layer**

Write Data Transformations

Write Data Transformations

Check Quality of Bronze

**Docs & Version**

Data Documenting

Versioning In GIT

**Validation**

Data Correctness Check

**Coding**

Data Cleansing

Explore and understand the Data

**Analyzing**

**Explore and Understand the Data**

* Crm\_cust\_info contain details and description about customers.
* Crm\_prd\_info contain current and historical data related to products.
* Crm\_sales\_details contains columns to join and it is a transactional table for sales
* Erp\_cust\_az12 contains extra details about customers
* Erp\_loc\_a101 is also about extra customer details
* Erp\_px\_cat\_g1v2 contain extra details about product

**DDL for silver layer**

* Same as bronze layer
* **Meta Data Columns**

Extra column added by data engineer that do not originate from the source system. We use is to provide extra information about each record.

1. Create\_date : The record’s load timestamp
2. Update\_date : The record’s last update timestamp
3. Source\_system : The origin system of the record
4. File\_location : The file source of the record

If you have data issue in the data warehouse(Corrupt data), this can help you to track exactly where this issue happen and when

**Clean and load crm\_cust\_info table**

* **A screenshot of a computer

  AI-generated content may be incorrect.**Before writing any data transformation or cleansing, we need to detect quality issue in the bronze layer. Without knowing the issue we cannot find the solution.
* Quality Check : Primary key must be unique and not null
* Quality Check : Check for unwanted spaces in string values
* Quality Check: Check the consistency of values in low cardinality columns.

In our data warehouse, we aim to store clear and meaningful value rather than using abbreviated terms.

In our data warehouse, we use the default value ‘n/a’ for missing values

**Clean and load crm\_prd\_info**

* In this table there is no problem with duplicate primary keys
* **A screenshot of a computer program

  AI-generated content may be incorrect.**Filter out unmatched data after applying transformation
* We found some products keys that are not listed in the sales tables, so we found those products still not ordered by customers.

**A screenshot of a computer

AI-generated content may be incorrect.**

* Checked for unwanted space in the product name column but couldn’t find a one
* Checked for null values or negative values in prd\_cost column, found 2 values and replaced it 0
* In crm\_prd\_info table, we cannot have end dates less than the start date.

For this kind of a complex transformations in sql, typically narrow it down to a specific example and brainstorm multiple solution approaches.

1. Switch start and end dates

Data now make no sense, dates are overlapping. And start date should have a record

1. Derive end dates from the start date

End date = start date if the next record -1

To implement this we take a subset of data and test the function we implemented and apply it to the full query.

Based on the columns derived we have to modify the silver table

Transformations that we have performed

* Derived columns: Create new columns based on calculations and transformations of existing ones.
* Data Enrichment: Add new, relevant data to enhance the data set for analysis

**Clean and load crm\_sales\_details**

* Checked for the spaces in sls\_ord\_num. But couldn’t find.
* **A screenshot of a computer

  AI-generated content may be incorrect.**We are going to connect this table with crm\_cust\_info and crm\_prd\_info tables. For that we need to check any prd\_key or cust\_ids are missing

* Check the quality of data, because it is in integer format
* Checked for invalid dates, and order of dates like order date is greater than ship date
* Now we need to check some business rules
* **Sales = Quantity \* Price**
* **Negative, zeros or null values are not allowed**
* **Solutions will be**

1. Data issue will be fixed directly in the source system
2. Data issues has to fixed on data warehouse

* **Rules**

1. If sales is negative, zero or null, derive it using quantity and price
2. If price is zero or null, calculate it using sales and quantity
3. If price is negative, convert into positive

**Clean and load erp\_cust\_az12**

* In here we are using it’s cid to connect with crm\_cust\_info. But there is a little bit difference.
* Identify out of range birth days

**Clean and Load erp\_loc\_a101**

* Had to remove “ – “ from the cid column

**Clean and Load erp\_px\_cat\_g1v2**

**Building Gold Layer**

Rename to friendly names

Choose Type

Dimension or Fact

Build the business Objects

**Docs & Version**

Data Documenting

Versioning In GIT

**Validation**

Data Integration Check

**Coding**

Data Integration

Explore and understand the Business Objects

**Analyzing**

**\**

**What is Data Modeling?**

Usually the source system deliver raw data which is unorganized, messy not very useful in current states. Now the data modeling is the process of taking this raw data and organize it and structure it in meaningful ways. We are putting the data in a new friendly and easy to understand objects like customers, orders, product each one of them is focused on specific information.

Conceptual Data Model

* The focus is only on the entity and their relationship between them

.

Logical Data Model

* Here we star specifying what are the different columns that we can in each entity.

Physical Data Model

* Here we have to add all technical details like adding for each column the data type and the length of each data type and many other details

**Star schema and Snowflake Schema**

**Star Schema**

* It has fact table in the central and surrounded by dimension tables. Fact table contains transactions, events while dimension table contains descriptive information. The relationship between fact table in the middle and dimension table around it forms like a star shape.

**Snowflake Schema**

* It is similar to star schema, but dimensions are divided into sub sections. Shape of this model look like snowflake

Star schema look easier, easy to understand, easy to query really perfect for analysis. Issue might be dimension might contain duplicates and dimensions get bigger with the time.

In a snowflake schema is more complex. need lots of knowledge and efforts in order to query something from snowflake. Advantage here comes with the normalization as you are breaking those redundancies in small tables.

**Dimensions and Fact tables**

* Dimensions contains descriptive information or like categories, that gives some context to your data. For example a product info you have product name, product category, sub category. This is like a table that is describing product .
* Facts are like events, transactions. They contain 3 important information

1. You have multiple ids (Multiple Products) from multiple dimensions
2. Date information
3. Measures and numbers

If you see those 3 types of data in one table, then this is a fact

If you have a table that answers, How much, How many then this is a fact

If you have a table that answers who, what, where then this is a dimension.

**Create Dimension Customer**

* To create this table, we join silver.crm\_cust\_info table with silver.erp\_cust\_az12, and silver.erp\_loc\_a101 using a left join. But the problem is when we have multiple joins we would get duplicate. Because relationship between those tables is not clear one to one , we might get one to many or many to many relationships

**A screenshot of a computer program

AI-generated content may be incorrect. After joining table, check of any duplicates were introduced by the join logic.**

* In here we can see two gender columns. One is from crm system and other one is from erp system. Now we have to perform data integration problem.

**A screenshot of a computer

AI-generated content may be incorrect.** Thers is a null values because customer who is in crm table not listed in erp table.

We consider crm table as master table

**Surrogate Key**

System generated unique identifier assigned to each record in the table

* In the product table we have historical data but in requirement they ask for current data not the historical data.
* Use dimension’s surrogate keys instead of IDs to easily connect fact with dimensions.