# 1. Introduction to DBT

DBT (Data Build Tool) is an open-source command-line tool that enables data analysts and engineers to transform data in their warehouse more effectively.

At its core, DBT is a transformation tool that allows you to write modular SQL queries to transform raw data into analytics-ready data models.

It brings software engineering practices like version control, testing, and documentation to data transformation work.

DBT doesn't extract or load data; it focuses exclusively on the transformation step in the ELT (Extract, Load, Transform) process.

## The key features that make DBT powerful include:

- **Version control integration**: All DBT code can be version controlled using Git
- **Testing and documentation**: Built-in functionality for testing and documenting your data transformations
- **Modular SQL**: Write reusable SQL code that follows software engineering best practices
- **Dependencies management**: Automatically handle the dependencies between different data models

## Why DBT?

Before DBT, data transformation work was often done through:

1. **SQL scripts**: Manually maintained, often with no version control
2. **ETL tools**: Complex and expensive tools that required specialized knowledge
3. **Custom code**: Python, Scala, or other programming languages used to transform data

These approaches led to several challenges:

- **Maintainability issues**: Hard to update and understand complex SQL scripts
- **Lack of testing**: No standardized way to test data transformations
- **Poor documentation**: Difficult to understand what transformations were doing

- **Dependency management**: No clear way to handle dependencies between transformations
- **Collaboration difficulties**: Hard for teams to work together on data transformations

DBT addresses these challenges by providing:

1. **Modularity**: Break down complex transformations into manageable pieces
2. **Reusability**: Create and reuse common SQL patterns across your project
3. **Testing framework**: Built-in testing capabilities to ensure data quality
4. **Documentation generation**: Automatically generate documentation for your data models
5. **Dependency management**: Automatically handle the order of model execution
6. **Development workflow**: Support for development environments separate from production

Organizations that adopt DBT typically see:

- Faster development cycles for data transformations
- Improved data quality through testing
- Better collaboration among data team members
- Clearer understanding of data lineage
- More reliable data pipelines