Agile Data Warehousing with SQL Server 2012

Davide Mauri
SolidQ

24 HOURS OF PASS

Global Sponsors:

Microsoft SQLSENTRY rssbus

sp_help ‘Davide Mauri’

Microsoft SQL Server MVP

Works with SQL Server from 6.5

Works on BI from 2003

Specialized in Data Solution Architecture, Database Design, Performance Tuning, Business Intelligence, Data Quality

President of UGISS (Italian SQL Server UG)

Author & Speaker in all major SQL Server conferences

Mentor @ SolidQ (Italy)

Agenda

Why a Data Warehouse is needed?

Agile Principles

Data Warehouse Modeling 101

Dimensions

Facts

Loading the Data Warehouse (overview)

Data Unit Testing

Why a Data Warehouse is needed?

The Data

Is scattered all around the field

Is stored in a way that is useful for application to be used and is not optimized for human usage

Is stored to optimize read and write access

It carries same information is stored in different ways

(e.g.: Gender: M/F or 1/2)

The User

Need to gather and analyze data autonomously

Need to correlate data from different data sources

Need to be able to get historical data

Need to work on trusted data

Data Analysis

Must not interfere with production systems

Has to be consistent over time

May be done using additional external data

Has to give only and only one trusted result

The Data Warehouse

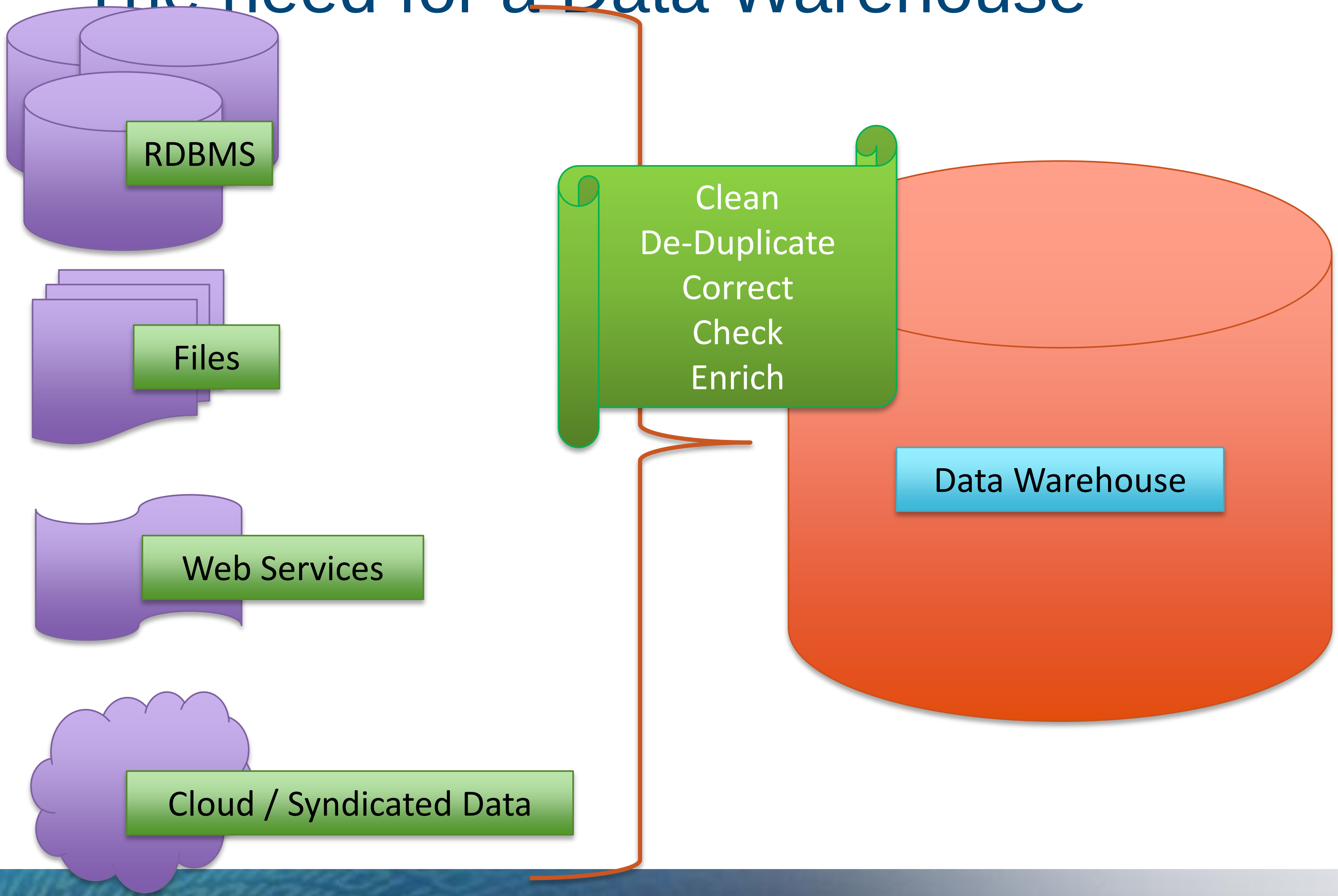
Is the solution to all the mentioned points

Is the central repository of all enterprise information

It provides *certified, cleansed, correct and user friendly* information

It has the ***single version of truth*** to which everyone in the company can refer to when needed

The need for a Data Warehouse



Myths and Fables

Data Warehouses take years to be developed

Data Warehouses can now be avoided since current s/w and h/w allows to directly connect to source data

Debunking a myth

«Data Warehouses take years to be developed»

TRUE.

But this doesn’t mean that they can deliver value just at the end of the development!

We need to change the way we build a DW, using an interative, evolutionary approach

Debunking a myth

«Data Warehouses can now be avoided»

Yes, technically is possible!

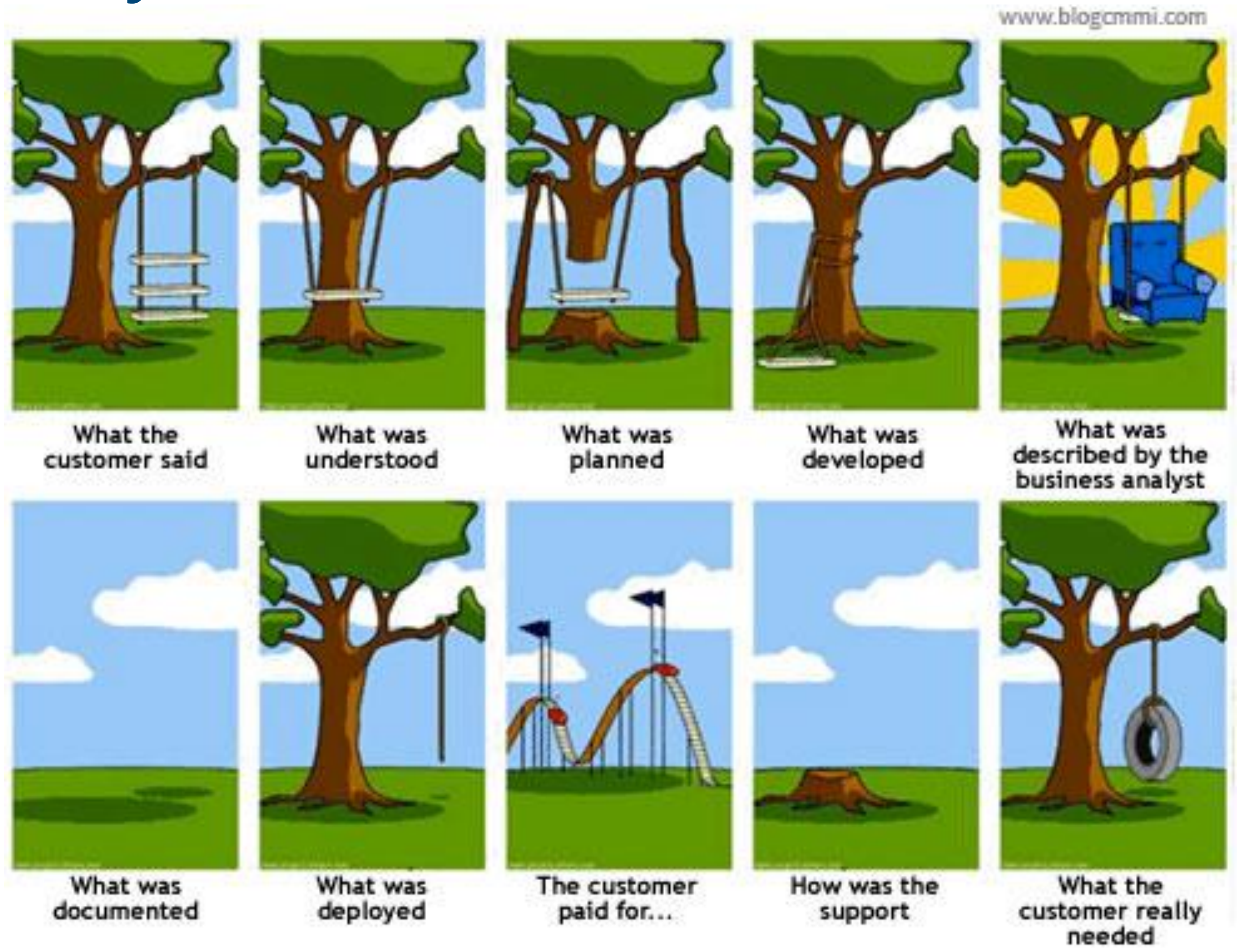
Well, technically is also possible to just but the parts of a car and built it yourself ☺

Ever tried to connect directly to a SAP database?

The truth is that a good DW is **vital!**

Agile Principles

The story so far



Agile Principles

Small design upfront. Prototype.

Delivery quickly, Deliver frequently

Users are part of the development team!

- Feedback is a key part of the success
- Their will grow with the solution and the solution will grow with them

Embrace Changes!

Agile Challenges

«Delivery Quickly and Fast»

Challenge: keep high quality, no matter who’s doing the work

«Embrace Changes»

Challenge: don’t introduce bugs. Change the smallest part possible. Use automatic Unit Testing to preserve and assure data quality.

Engineering the solution

To be Agile, some engineering practices needs to be included in our work model

- Apply well-known models
- Define rules
- Automate and/or Check rules application
- Measure
- Test

Data Warehouse Modeling 101

The need for a Data Warehouse

DataMart or Data Warehouse ?

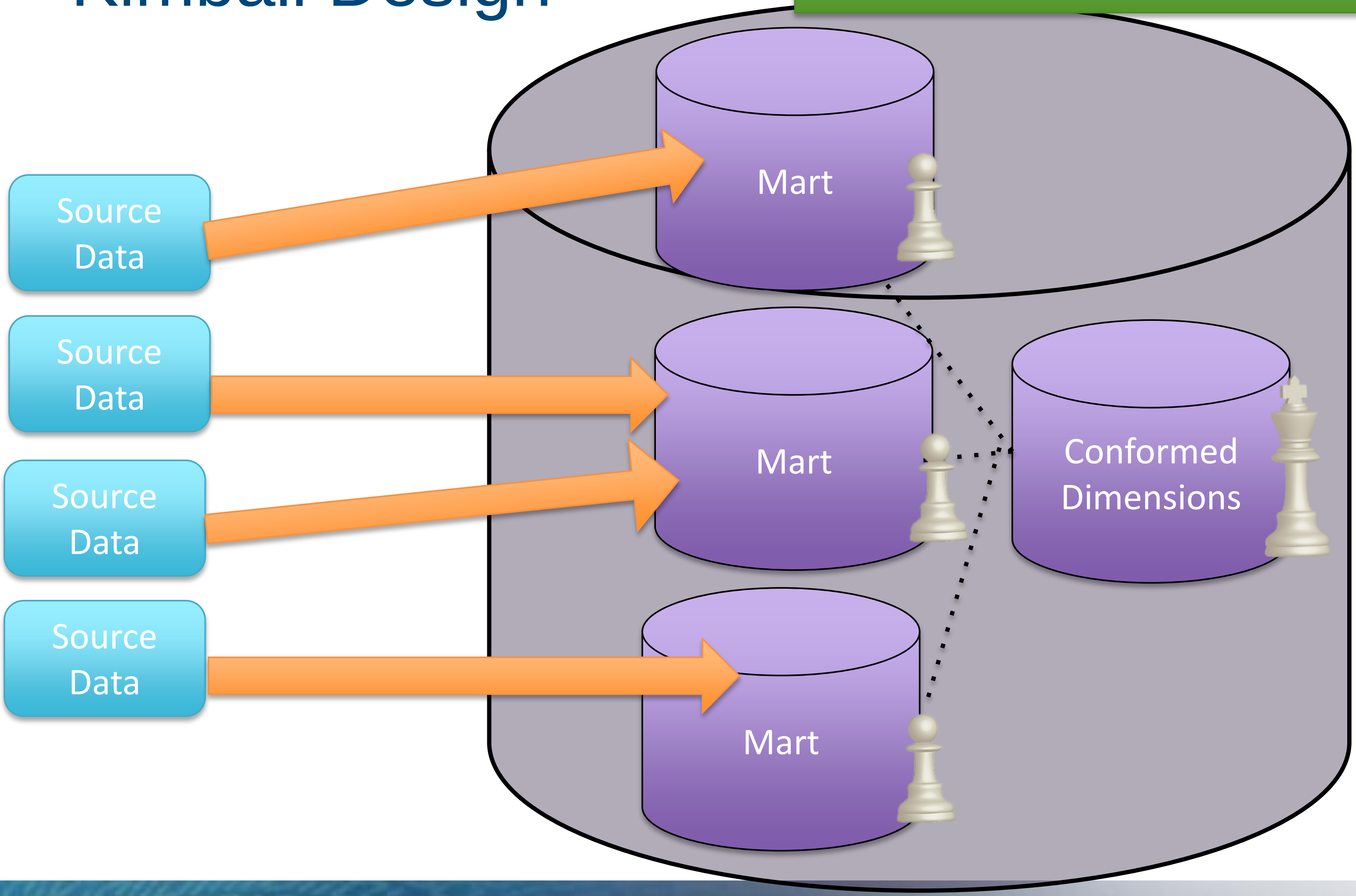
- No “Standard” definition, but usually
  - «datamarts» contains departmental data
  - «Data Warehouse» contains all data
- Depends of the approach used
  - Inmon
  - Kimball
- Other approaches available, but the two mentioned here are the most well-know and adopted right now

Inmon or Kimball ?

- Both have pro and cons
  - Of course the difference between the two is not only limited to the Data Warehouse definition!
- Why not both? ☺
  - Avoid religion wars and take the best of both worlds☺

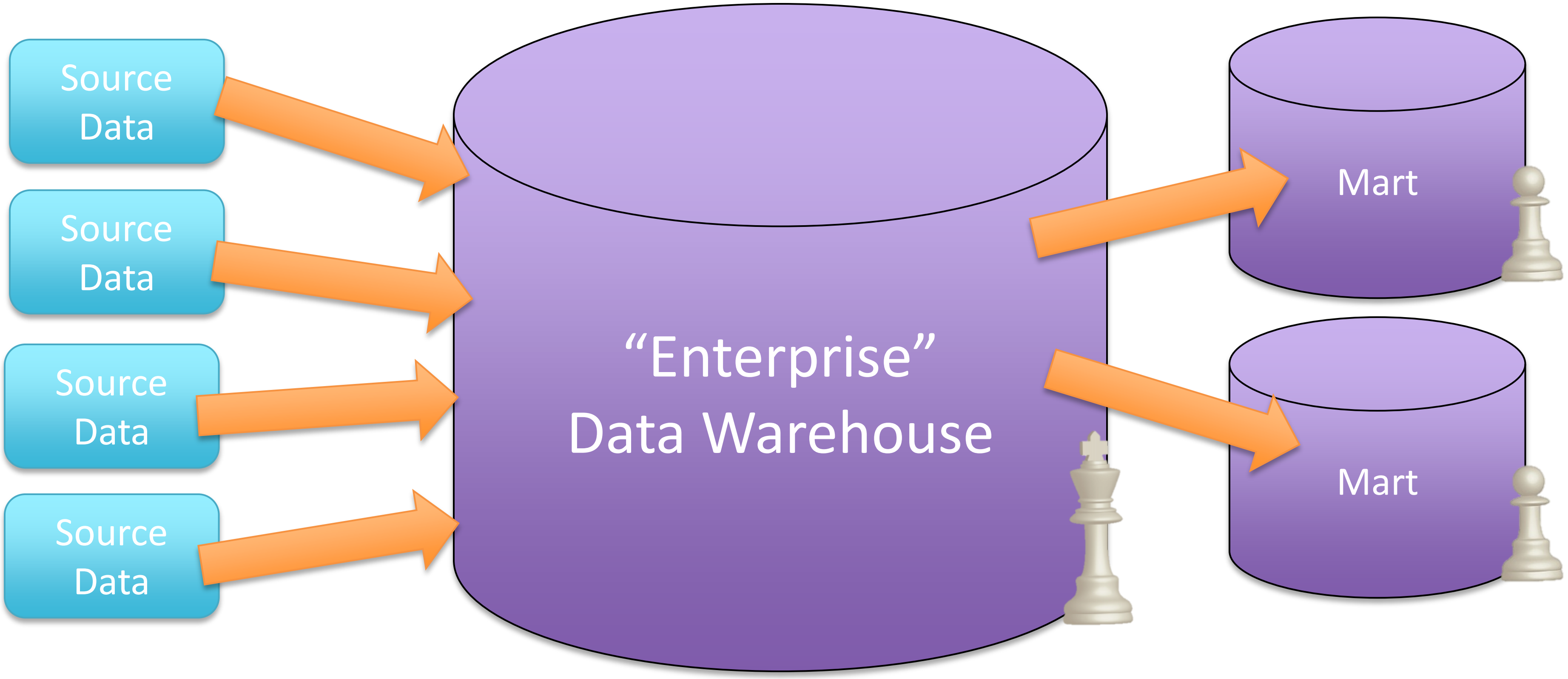
Kimball Design

Data Warehouse ::= is the sum of all Data Marts and the conformed dimensions



Inmon Design

Data Warehouse ::= THE corporate wide data model
Datamarts :== Subsets of the Data Warehouse



The need for a Data Warehouse

On average, Kimball approach is the most used

- Easy to understand, Easy to use
- Efficient, Well supported by tools
- Well known

But the idea of having one physical DWH is very good

- Again, the advice is not to be too rigid
- Be willing to mix the things of move from one to another
- My «Perfect Solution» is an Inmon Datawarehouse used to generate Kimball Datamarts

Use the design you prefer but expose star-schema data

- Use **views** to create an **abstraction layer**

The need for a Data Warehouse

Data Warehouse may still not offer good performance

- Due to huge amount of data
- Due to complex queries

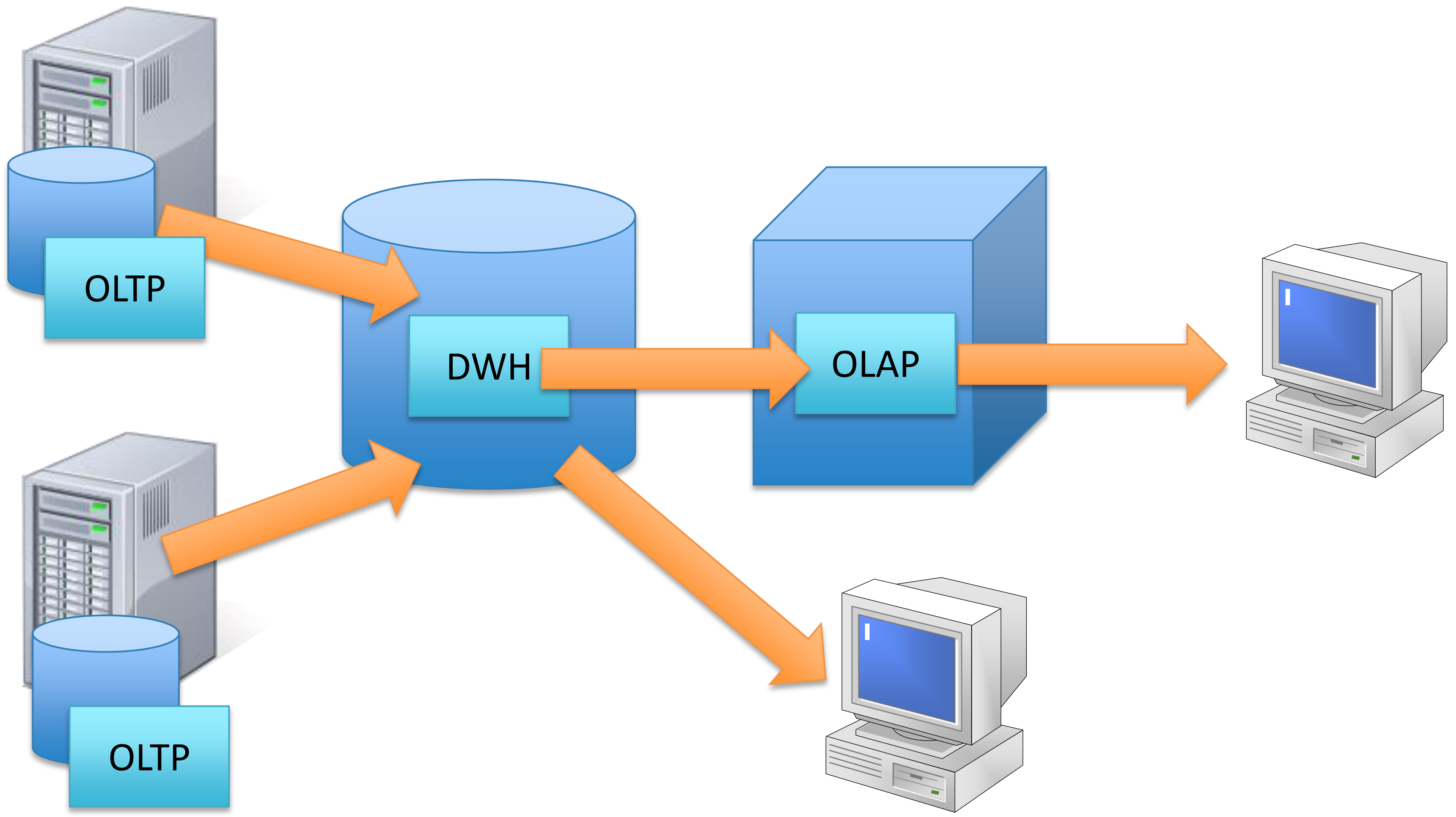
Why this happens?

- Data is usually stored with the highest level of detail in order to allow any kind of analysis
- User usually needs aggregated data

One solution is to move away from RDBMS for querying

- Use a system specifically created for this
- Another solution is to stay with RDBMS but optimize it for this purpose (Indexed Views, Parallel Data Warehouse, Column Aligned Storage, …)

The Complete Picture



Basic Concepts

Dimensional Modeling

- Modeling a database schema using “Facts” and “Dimension” entities
- Proposed and Documented by Kimball (1997)
- Applicable both to Relational and Multidimensional database
  - SQL Server
  - Analysis Services
- Focus on the end user

Basic Concepts

A «fact» is something happened

- A product has been sold
- A contract has been signed
- A payment has been made

Facts contains «measurable» data

- Product final price
- Contract value
- Paid Amount

The measurable data is called a «Measure»

Basic Concepts

Measures are usually «Additive»

- Make sense to sum up measure values
- Eg: money amount, quantity, etc.

«Semi-Additive» data exists

- Data that cannot be summed up
  - Eg: Account balance
- Tools may have specific support for semi-additive measures
  - Analysis Services has it ☺

Basic Concepts

Dimensions define how facts can be analyzed

- Provide a meaning to the fact
  - Categorize and classify the fact
- Eg: Customer, Date, Product, etc.

Dimensions have attributes

- Attributes are the building block of a Dimensions
- Eg: Customer Name, Customer Surname, Product Color, etc..

Dimensions are stored in «Dimension Tables»

- “Dimension Members” are the values stored in Dimensions

Basic Concepts

Dimension Modeling come in two flavors

- Star Schema
- Snowflake Schema

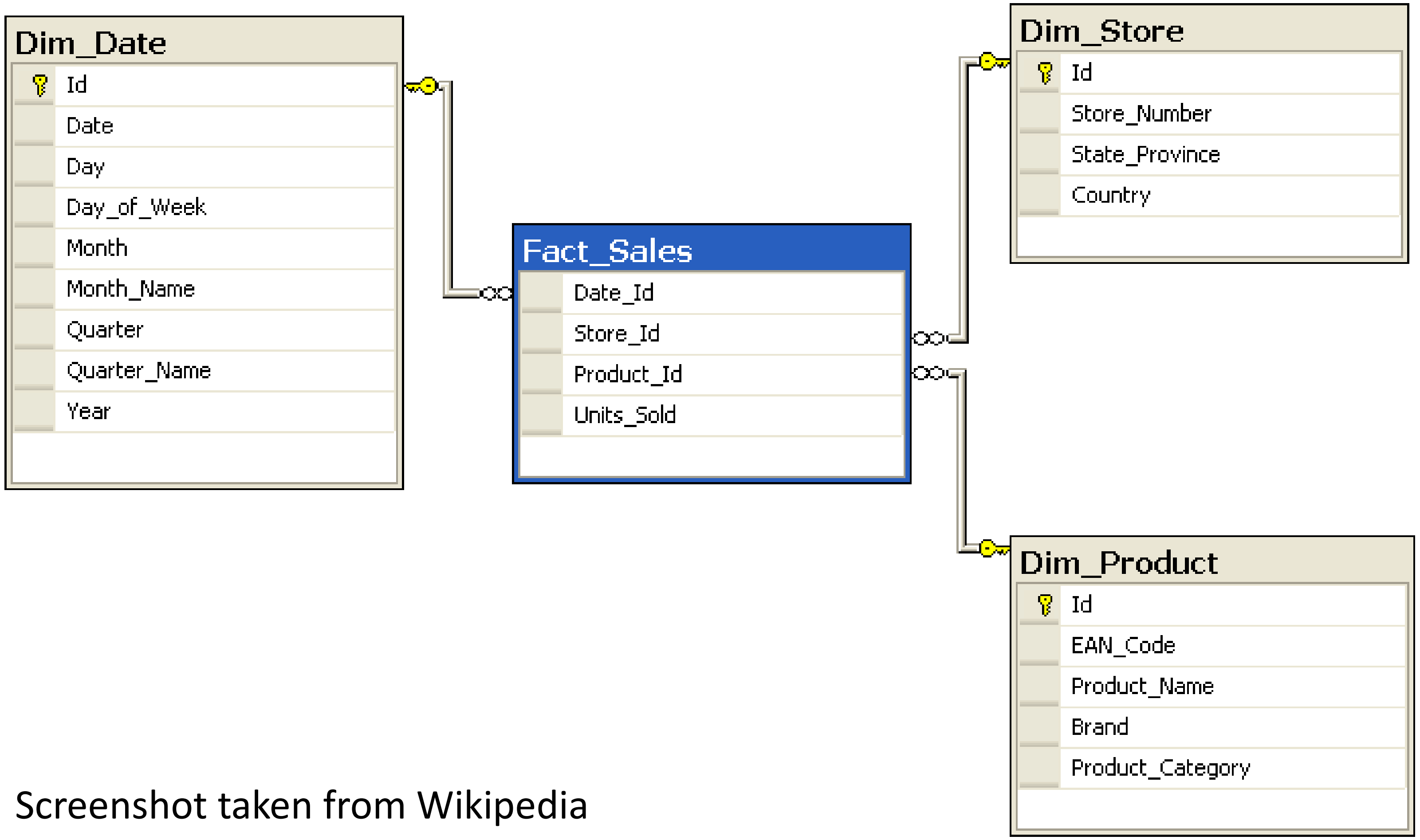
Star Schema

- Dimensions have direct relationship with fact tables

Snowflake Schema

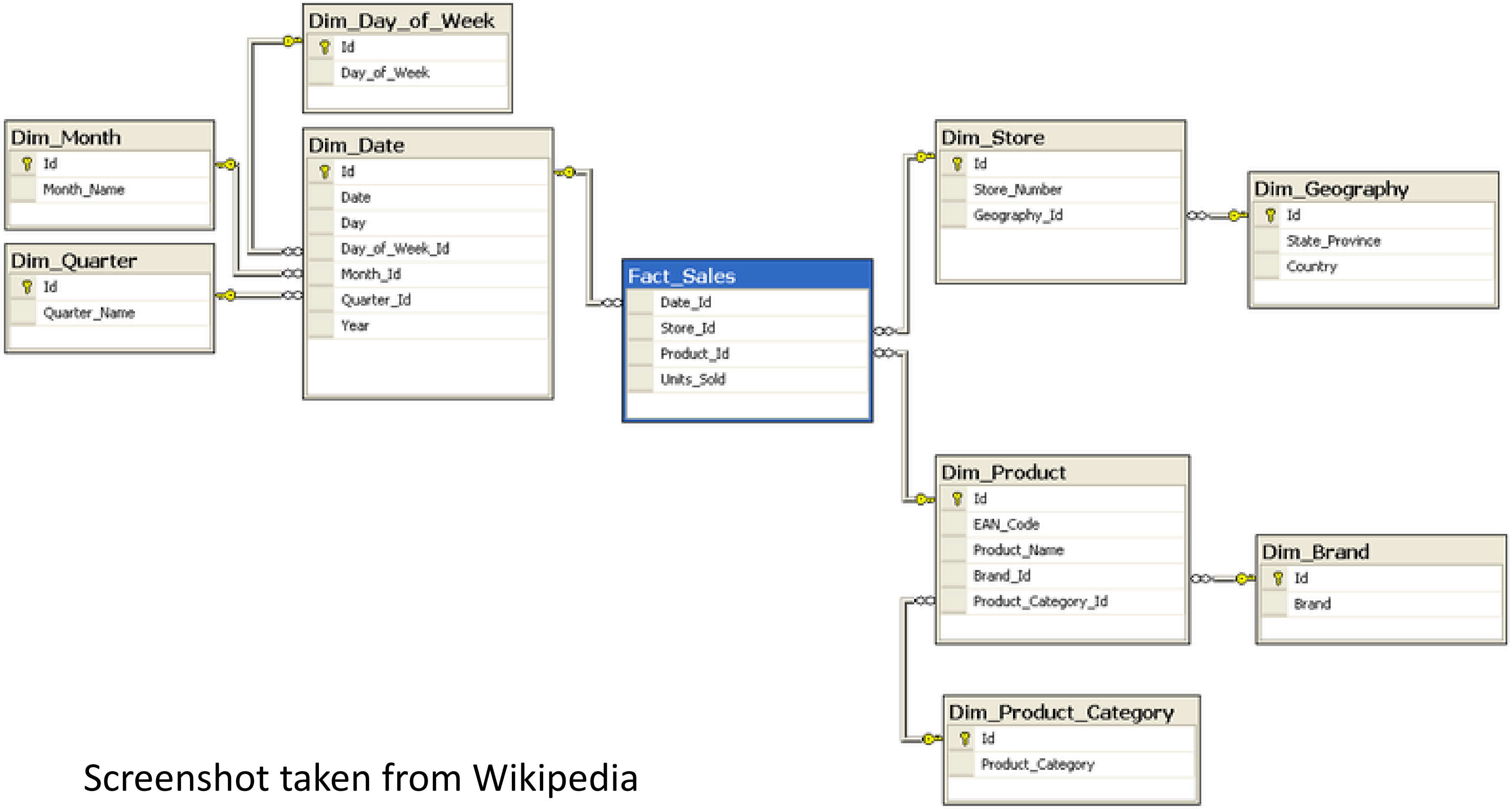
- Dimension may have an indirect relationship with fact fables

Basic Concepts – Star Schema



Screenshot taken from Wikipedia

Basic Concepts – Snowflake Schema



Screenshot taken from Wikipedia

Dimensions

Dimensions Design Patterns

Design Pattern «Slowly Changing Dimensions»

- Introduced by Kimball

Define how to deal with changing data in attributes

Three well-known pattern

- But there can be many more!

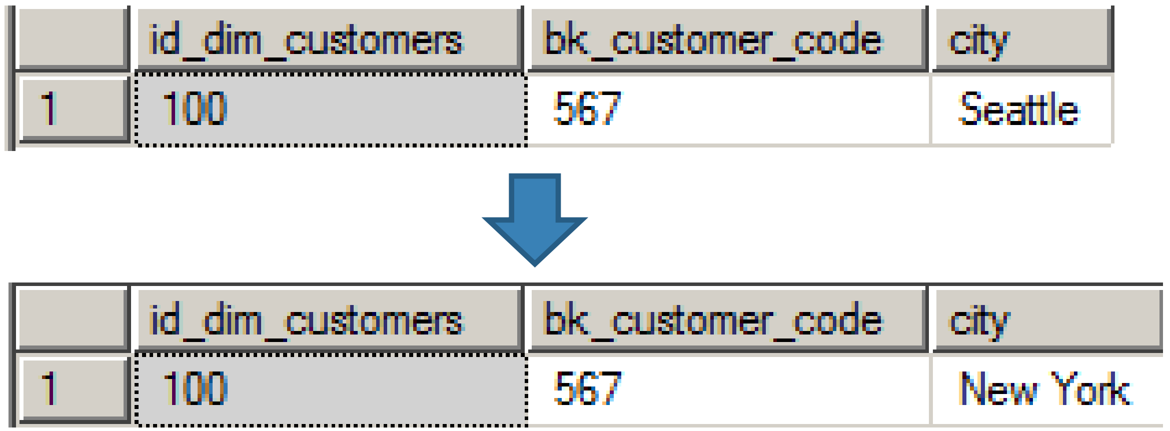
SCD Type 1

Scope

Update data to last value

Implementation

UPDATE



	id_dim_customers	bk_customer_code	city
1	100	567	Seattle

	id_dim_customers	bk_customer_code	city
1	100	567	New York

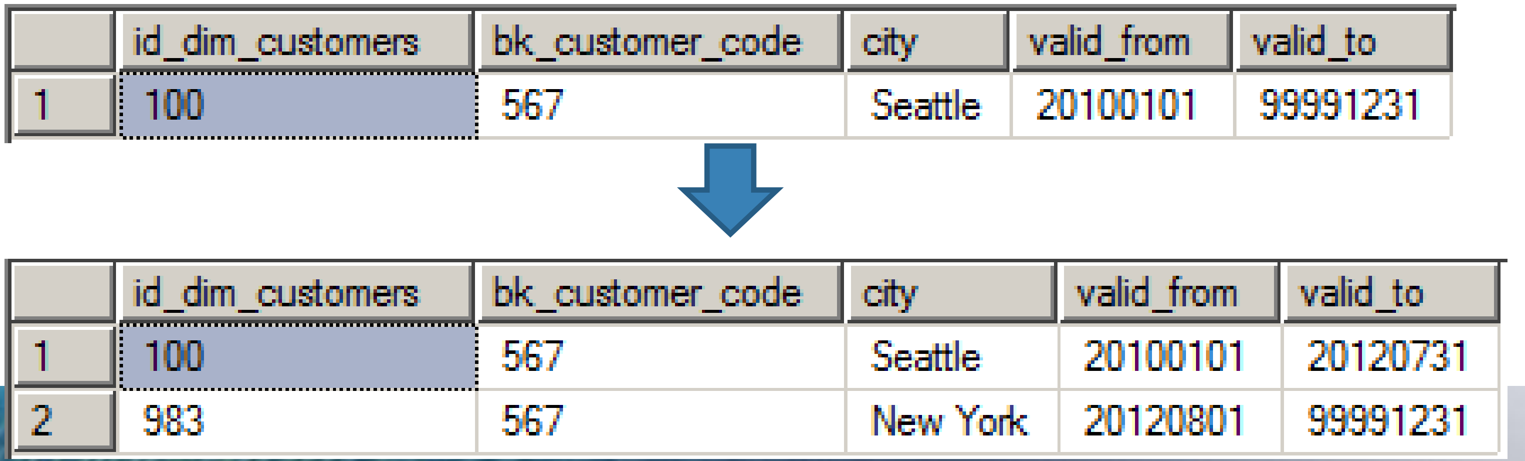
SCD Type 2

Scope

Keep the all the past values and the current ones

Implementation

Row Valid Time + UPDATE + INSERT



	id_dim_customers	bk_customer_code	city	valid_from	valid_to
1	100	567	Seattle	20100101	99991231

	id_dim_customers	bk_customer_code	city	valid_from	valid_to
1	100	567	Seattle	20100101	20120731
2	983	567	New York	20120801	99991231

SCD Type 3

Scope

Keep the current value and the one before that only

Implementation

Specific Columns + UPDATE

Basic Dimension Rules

Must have a specific, integer, source-independent, artificial key

Source PK columns are stored inside and prefixed with bk_

- Business Keys

Must have specific columns to store (at least)

- last change date
- scd_checksum1/scd_checksum2
  - MD5 algorithm at least

Facts

Fact Tables

Fact tables are related to dimensions using dimension id only

Data should be stored at the highest detail (granularity)

- Aggregations will be done
  - Later
  - On The Fly
  - In-Memory
  - With specific solutions (Multidimensional DBs)

Different fact tables (with different granularity) may exists and may (should) use the same dimensions if needed

Transactional Fact Table

Used to store «Transactional Data»

- Sales
- Invoices
- Quantities

Each row represent an event happened in a specific point in time

Snapshot Fact Table

Useful when you need to store inventory/stock/quotes data

- Data that is *not* additive

Store the entire situation of a precise point in time

- «Picture of the moment»

Expensive in terms of data usage, usualy snapshot are at week level or above (months / semester ecc..)

Temporal Snapshot Fact Table

New approach to store snapshot data without doing snapshots

- Each rows doesn’t represent a point in time but a time interval

Bring Temporal Database theory into Data Warehousing

Allows the user to have daily (or even hourly) snapshot of data

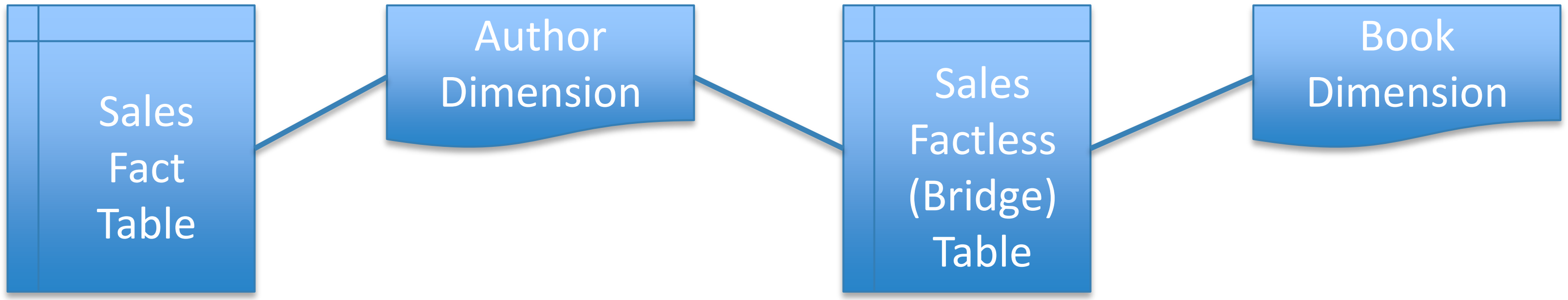
- Avoiding data explosion

Look in the PASS 2011 DVDs or SQL Bits X website

Bridge / Factless Tables

Sometimes the whole is not made of the sum of the single elements.

Imagine a Bookstore:



The bridge table doesn’t contain facts…so it’s a «factless» table. It’s only used to store M:N relationship.

Loading the Data Warehouse

Loading the Data Warehouse

- The loading phase is called ETL
  - Extract
  - Transform
  - Load
- In this phase data is loaded from sources, cleansed, conformed, checked, enriched and re-shaped into the Data Warehouse schema

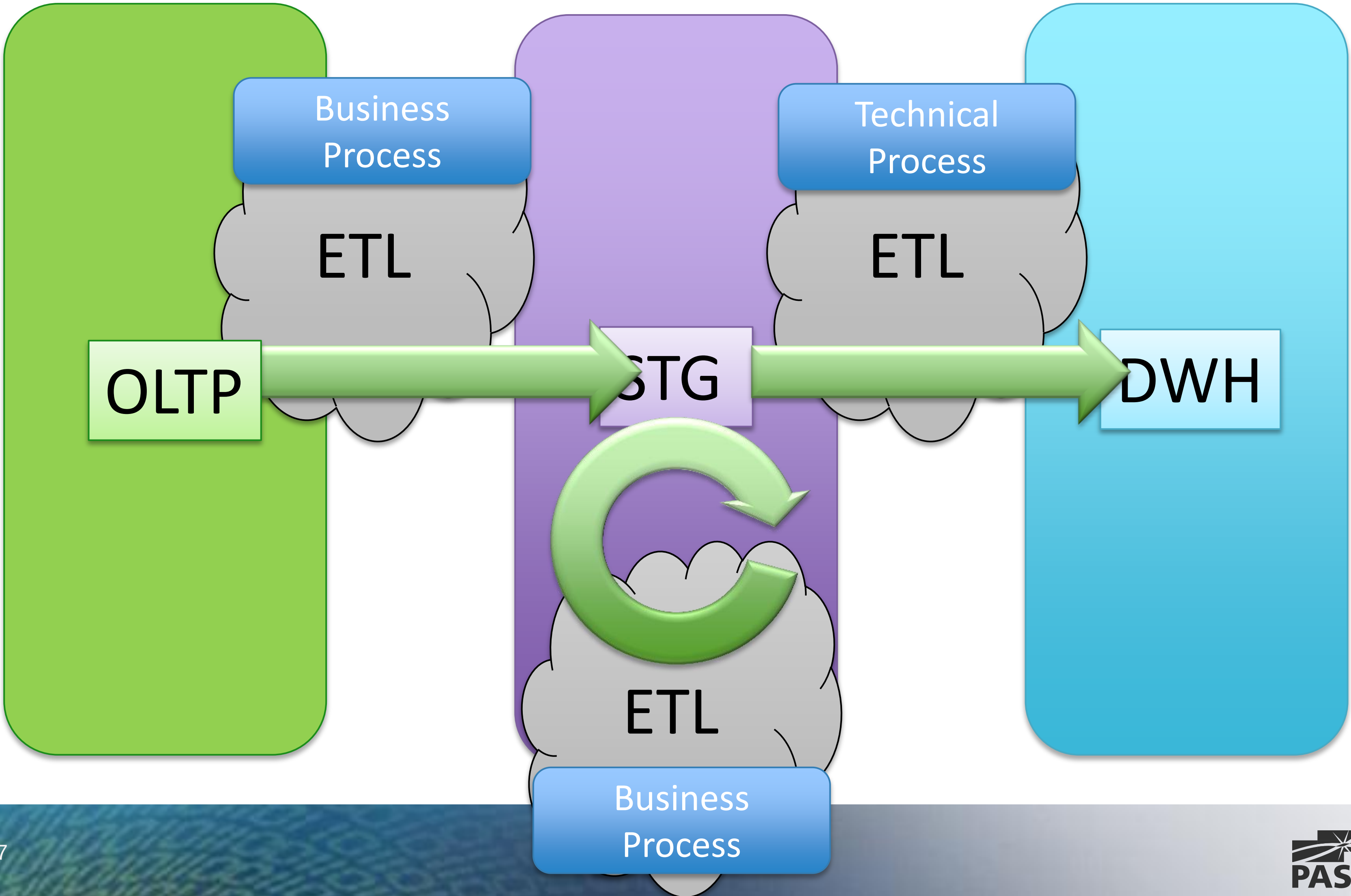
Loading the Data Warehouse

To be able to be Agile is *vital* to keep business and technical process completely separated

Business Process: ETL logic that can be applied only to the specific solution you're building

Technical Process: ETL logic that can be used with *any* Data Warehouse and that can be highly automated

Loading the Data Warehouse



Testing

Data Warehouse Unit Test

Before releasing *anything* data in the DW must be tested.

User has to validate a sample of data

(e.g.:total invoice amount of January 2012)

That validated value will become the reference value

Before release, the same query will be executed again. If the data is the expected reference data then test is green otherwise the test fails

Data Warehouse Unit Test

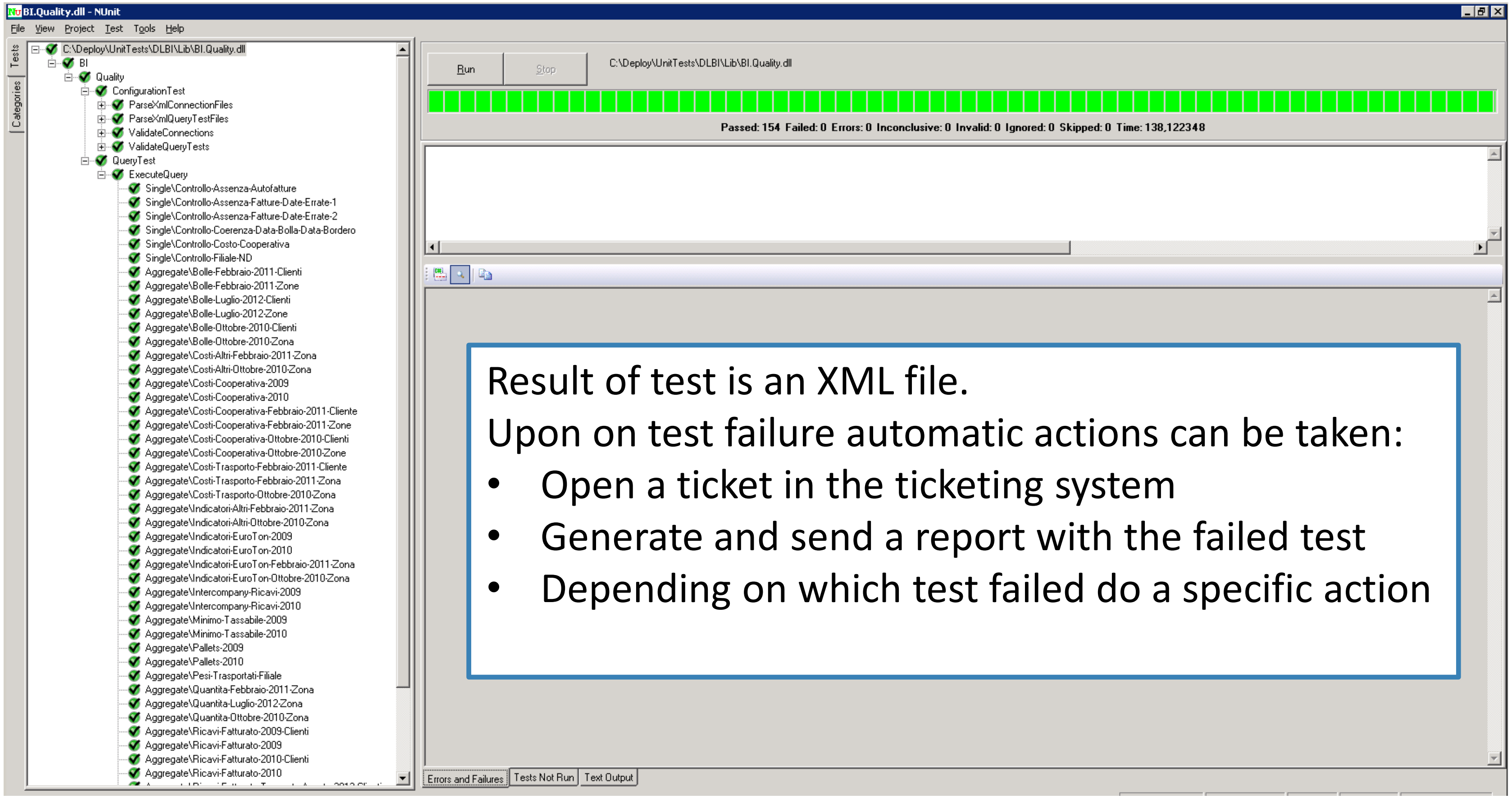
Of course test MUST be automated when possibile

- Visual Studio
- BI.Quality (on CodePlex)
  - Based on Nunit

What to test?

- Aggregated results
- Specific values of some «special» rule
- Fixed bugs/tickets

Data Warehouse Unit Test



```
C:\Deploy\UnitTests\DLBI\Lib\BI.Quality.dll
  BI
    Quality
      ConfigurationTest
        ParseXmlConnectionFiles
        ParseXmlQueryTestFiles
        ValidateConnections
        ValidateQueryTests
      QueryTest
        ExecuteQuery
          Single\Controllo-Assenza-Autofatture
          Single\Controllo-Assenza-Fatture-Date-Errate-1
          Single\Controllo-Assenza-Fatture-Date-Errate-2
          Single\Controllo-Coerenza-Data-Bolla-Data-Bordero
          Single\Controllo-Costo-Cooperativa
          Single\Controllo-Filiale-ND
          Aggregate\Bolle-Febbraio-2011-Clienti
          Aggregate\Bolle-Febbraio-2011-Zone
          Aggregate\Bolle-Luglio-2012-Clienti
          Aggregate\Bolle-Luglio-2012-Zone
          Aggregate\Bolle-Ottobre-2010-Clienti
          Aggregate\Bolle-Ottobre-2010-Zona
          Aggregate\Costi-Altri-Febbraio-2011-Zona
          Aggregate\Costi-Altri-Ottobre-2010-Zona
          Aggregate\Costi-Cooperativa-2009
          Aggregate\Costi-Cooperativa-2010
          Aggregate\Costi-Cooperativa-Febbraio-2011-Cliente
          Aggregate\Costi-Cooperativa-Febbraio-2011-Zone
          Aggregate\Costi-Cooperativa-Ottobre-2010-Clienti
          Aggregate\Costi-Cooperativa-Ottobre-2010-Zone
          Aggregate\Costi-Trasporto-Febbraio-2011-Cliente
          Aggregate\Costi-Trasporto-Febbraio-2011-Zona
          Aggregate\Costi-Trasporto-Ottobre-2010-Zona
          Aggregate\Indicatori-Altri-Febbraio-2011-Zona
          Aggregate\Indicatori-Altri-Ottobre-2010-Zona
          Aggregate\Indicatori-EuroTon-2009
          Aggregate\Indicatori-EuroTon-2010
          Aggregate\Indicatori-EuroTon-Febbraio-2011-Zona
          Aggregate\Indicatori-EuroTon-Ottobre-2010-Zona
          Aggregate\Intercompany-Ricavi-2009
          Aggregate\Intercompany-Ricavi-2010
          Aggregate\Minimo-Tassabile-2009
          Aggregate\Minimo-Tassabile-2010
          Aggregate\Pallets-2009
          Aggregate\Pallets-2010
          Aggregate\Pesi-Trasportati-Filiale
          Aggregate\Quantita-Febbraio-2011-Zona
          Aggregate\Quantita-Luglio-2012-Zona
          Aggregate\Quantita-Ottobre-2010-Zona
          Aggregate\Ricavi-Fatturato-2009-Clienti
          Aggregate\Ricavi-Fatturato-2009
          Aggregate\Ricavi-Fatturato-2010-Clienti
          Aggregate\Ricavi-Fatturato-2010
```

```
C:\Deploy\UnitTests\DLBI\Lib\BI.Quality.dll
Passed: 154  Failed: 0  Errors: 0  Inconclusive: 0  Invalid: 0  Ignored: 0  Skipped: 0  Time: 138,122348
```

Result of test is an XML file.

Upon on test failure automatic actions can be taken:

- Open a ticket in the ticketing system
- Generate and send a report with the failed test
- Depending on which test failed do a specific action

Questions?

24 HOURS OF PASS

Global Sponsors:

Microsoft SQL SENTRY rssbus

Thank You for Attending

24 HOURS OF PASS

Global Sponsors:

Microsoft SQL SENTRY rssbus