

# **Relational Databases and Datawarehousing DWH + BI**

# What do we learn?

- Basic concepts, advantages and disadvantages of Data Warehousing
- Differences between OLTP and Data Warehousing systems
- Architecture and most important components of a Data Warehouse
- How Data Warehousing has evolved
- Problems with Data Warehousing
- The concept of a data mart
- Activities to start a Data Warehouse project
- Kimball's methodology
- Concepts of Dimensional modelling
- Creation of a DWH

# Introduction

“We proberen continu op de hoogte te blijven. **Rapportering is daarbij zeer belangrijk. Ik weet elke dag hoeveel volk er in Pairi Daiza en SnowWorld is geweest, hoeveel dat is in vergelijking met het jaar ervoor, en hoeveel de mensen gespendeerd hebben.** Ik weet ook elke dag alles van elk van onze deelelementen in Durbuy, **tot en met hoeveel mensen er snoepjes kopen of mountainbikes gehuurd hebben.** Wij worden geroepen op speciale momenten, zoals een overname, een investering of personeelskeuzes. Dan willen wij ook toegevoegde waarde bieden, en **alleen als je continu al je dossiers beheerst, kan je er heel snel op inspelen.**”

*Marc Coucke in Trends op 22/01/2020*



# April 2021 Job opening:



**VISSE &  
VAN BAARS**  
HET BI & BIG DATA NETWERK

## SQL DWH Developer



ETL, DWH and reporting



N.O.T.K



Antwerpen



03 808 1551 – 0490 64 33 50



[mSPIJker@visservanbaars.be](mailto:mSPIJker@visservanbaars.be)

*Voor een klant van mij in Antwerpen ben ik opzoek naar een SQL Developer die ervaring heeft in SQL Server, SSIS, DWH en reporting*

- SQL databanken, -queries
- DWH architectuur
- Reporting
- Power BI
- Sterk communicatief NL

# Introduction

- There is a growing need to turn data into information
  - Flexible business reporting available for the business
  - Data in DWH grows exponentially
    - Terabytes of data in a DWH are 'normal'
  - Applications using data have become more complex
    - Traditional reporting
    - Advanced analysis
  - All traditional DBMS offer DWH facilities

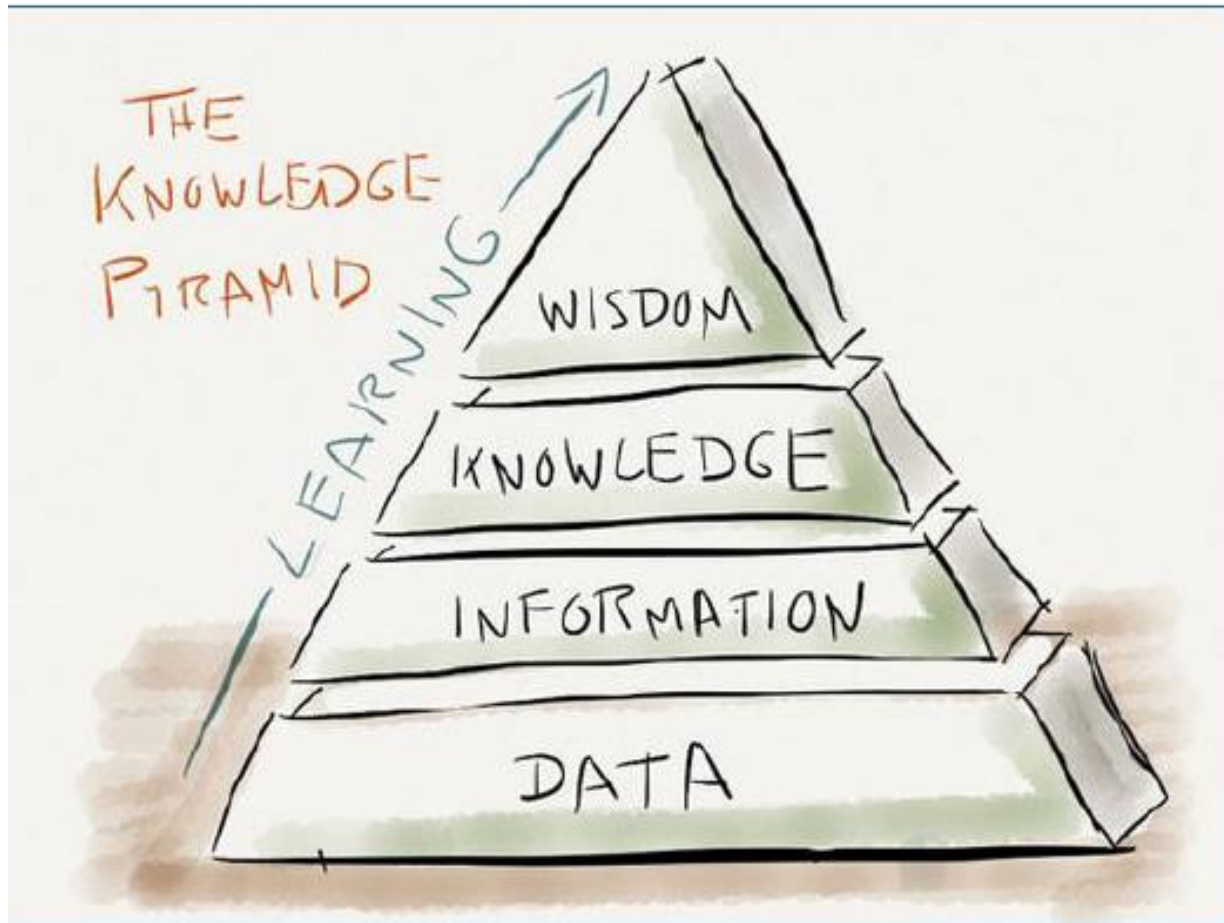


Image: pinterest



# Business Intelligence (BI): definition

Business Intelligence(BI) comprises the set of [strategies](#), [processes](#), [applications](#), [data](#), [technologies](#) and [technical architectures](#) which are used by enterprises to support the collection, data analysis, presentation and dissemination of business information.

BI technologies provide [historical](#), [current](#) and [predictive views of business operations](#).

Common functions of business intelligence technologies include [reporting](#), [online analytical processing](#), [analytics](#), [data mining](#), [process mining](#), [complex event processing](#), [business performance management](#), [benchmarking](#), [text mining](#), [predictive analytics](#) and [prescriptive analytics](#).

*Source: wikipedia*



# Business Intelligence (BI): another definition

- Business intelligence, or BI, is a blanket term for technology that helps companies organize and query their data to help them improve operations and make more money.

*Source: <http://fortune.com/2017/04/25/infor-buys-business-intelligence-player>*

# Drivers for increasing use of BI

- Digitalisation (ERP, CRM, PLM, DAM, PIM, ...)
- Connectors between BI software and Business software
  - Way of working can stay the same
  - BI comes on top of the existing SW
- Data overflow
- Complexity and Speed of change in the Business Environment
  - Gut feeling and experience is often not sufficient anymore
- Reduce inefficiencies, inaccuracies
  - Anti – Excel culture
- Decreasing Cost

# BI technology vendors

- Microsoft:
  - Reporting: Microsoft reporting services + PowerBI
  - BI + data mining: SSAS (SQL Server Analysis Services)
  - ETL: SSIS (SQL Server Integration Services)
- Cognos (now IBM)
  - ETL
  - Reporting tools
- Business Objects (now SAP)
  - Reporting
  - ETL
- SAP Business Warehouse
  - Kubus
- Tableau
- Datastage ETL
- QlikView rapportering



*Gartner Magic Quadrant for Business Intelligence and Analytics Platforms, March 2022 (commercial software)*

# Datawarehouse: definition

A **data warehouse** is an **integrated, subject oriented, time variant** and **non volatile** collection of data to support decisions taken on management level.

- **Subject oriented**

- The warehouse is organized around the major subjects of the enterprise (e.g. customers, products, and sales) rather than the major application areas (e.g. customer invoicing, stock control, and product sales).
- This is reflected in the need to store decision-support data rather than application-oriented data.

# Properties

- **Integrated**

- The data warehouse integrates corporate application-oriented data from different source systems, which often includes data that is inconsistent.

- **Time**

- Data in the warehouse is only accurate and valid at some point in time or over some time interval.
- Time-variance is also shown in the extended time that the data is held, the implicit or explicit association of time with all data, and the fact that the data represents a series of snapshots:
  - E.g. unit price of a product can be stored historically or can be a snapshot,
  - DW will built history by taking regular snapshots.

# Properties

- **Non volatile**
  - Data in the warehouse is not normally updated in real-time (RT) but is refreshed from operational systems on a regular basis. (However, emerging trend is towards RT or near RT DWs).
  - New data is always added as a supplement to the database, rather than a replacement.
- **Aggregated data**
  - E.g. generated by GROUP BY

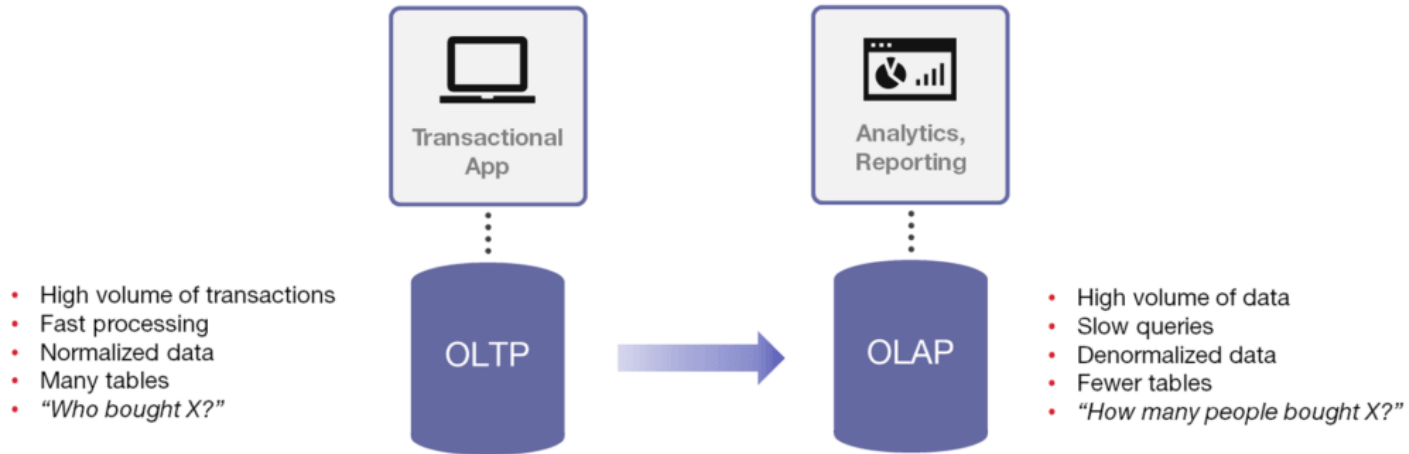
# Properties

- A DWH is just a regular relational database
- The difference with a 'regular' database is that the DWH is not specifically implemented for an application, but that it is a collection of data from all kinds of different source databases.
- Building a data warehouse gives you the opportunity to store the data in a structure that is suitable for quickly and adequately reading information from a database.



# OLTP vs OLAP database

## OLTP vs OLAP



# OLTP vs OLAP database

- OLTP = Online Transaction Processing  
[Online here has nothing to do with the internet and being 'online'. The term comes from the time when it was special that you could talk directly to a database. Online therefore stands for the fact that you have a real-time connection with the database and can work interactively with the database.]
- In OLTP the use of the database is characterized by many small actions, a relatively large number of which involve changing or adding data.
- OLTP databases are often normalized. Normalizing a database prevents redundancy and increases data consistency.
  - Writing to a database is faster if it has less redundancy.
  - The tables become 'small' => 1 record is small and easy to insert or update.

# OLTP vs OLAP database

- OLAP = OnLine Analytical Processing
- In OLAP the use of the database is characterized by mainly reading mostly larger data sets, often to make reports or to do analyses.
- OLAP workload does not benefit from normalization, rather the opposite. More tables tend to make writing queries for data analysis more complex and often slower (a.o. because of a lot of joins)
- Indexes make reading from a database faster but writing to the database slower. This rule implies that indexes are good for OLAP, but worse for OLTP. A reporting database can contain indexes that are specifically created for reporting and analysis. This provides improved performance compared to the OLTP database.

# Comparison of OLTP Systems and Data Warehousing

CHARACTERISTIC	OLTP SYSTEMS	DATA WAREHOUSING SYSTEMS
<b>Main purpose</b>	Support operational processing	Support analytical processing
<b>Data age</b>	Current	Historic (but trend is toward also including current data)
<b>Data latency</b>	Real-time	Depends on length of cycle for data supplements to warehouse (but trend is toward real-time supplements)
<b>Data granularity</b>	Detailed data	Detailed data, lightly and highly summarized data
<b>Data processing</b>	Predictable pattern of data insertions, deletions, updates, and queries. High level of transaction throughput.	Less predictable pattern of data queries; medium to low level of transaction throughput
<b>Reporting</b>	Predictable, one-dimensional, relatively static fixed reporting	Unpredictable, multidimensional, dynamic reporting
<b>Users</b>	Serves large number of operational users	Serves lower number of managerial users (but trend is also toward supporting analytical requirements of operational users)

# Why build a DWH?

- To solve performance related problems
  - Performance of queries in operational database is not good. Reporting and analyses queries are often slower when the operational database is normalized and the use of indexes is limited to the minimum.
  - Writing the right queries in operational database is difficult and takes a long time.
  - OLAP workload comprises OLTP performance if they need the same data at the same time
    - For analyses often large data sets are needed. Large data sets can push important data out of memory for the OLTP process, slowing down the OLTP.
    - Concurrency problems: 1 user wants to update 1 record that is being involved in a read operation due to an analysis => the transaction can't put an exclusive lock on this record

# Why build a DWH?

- To solve quality related problems
  - The quality of the data in an operational databases is not good  
[duplicate data → Duplicate records in operational systems lead to erroneous results during analysis / missing data / wrong data / inconsistent data]
  - Different operational databases contain information that does not match. If two different systems should give the same figures, in practice there will often be a difference between the two systems. One central DWH ensures the use of one central definition of terms such as turnover and cost-of-sales.
  - History is often lost in operational databases.

What happens  
when Jan moves  
to regio = Zuid?

VerkoperID	Verkoper	Regio
1	Jan	Noord
2	Piet	Zuid

VerkoperID	KlantID	Datum	Omzet
1	1	10-jun-2013	€ 1500,-
1	2	18-jun-2013	€ 2500,-
2	3	11-jun-2013	€ 1000,-
2	4	20-jun-2013	€ 750,-

```
SELECT Regio, SUM(Omzet) AS Totaal
FROM Verkoper JOIN Order
ON Verkoper.VerkoopID = Order.VerkoopID
GROUP BY Regio
```

# Goals of DWH

- Reporting
- Analysis of events in the past or actual events
- Predictions based on trend analysis
- Multidimensional reporting
- Empowerment of end user by offering simplified reporting tools (cf. SQL: only specialist can write SQL)
- Data mining

*DWH is a technology used to realize BI solutions,  
but it is by far not the only technology.*



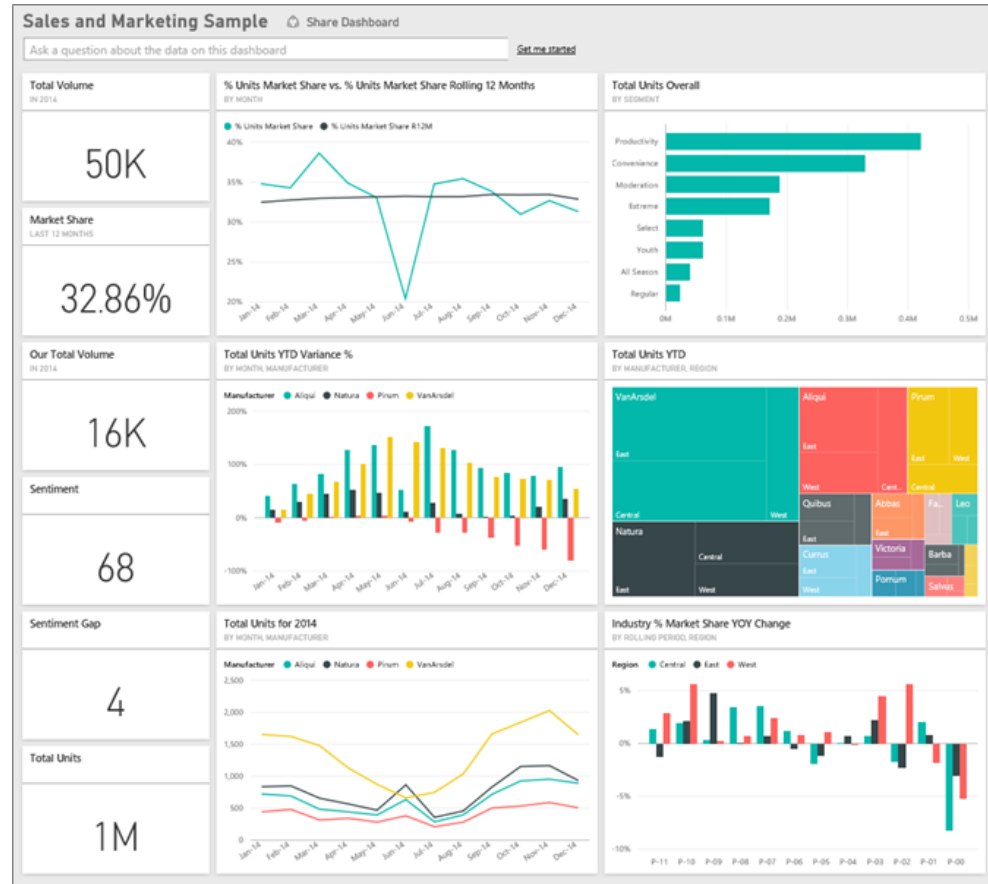
# Goals of DWH

- Reporting →  
Multidimensional report

Genre	2013	2014	2015	2016	2017	Total
Alternative		5.94	3.96	3.96		13.86
Alternative & Punk	62.37	39.60	45.54	38.61	55.44	241.56
Blues	10.89	10.89	19.80	8.91	9.90	60.39
Bossa Nova	0.99	1.98	7.92		3.96	14.85
Classical		13.86	9.90	16.83		40.59
Comedy		3.98	1.99	11.94		17.91
Drama		17.91	11.94	17.91	9.95	57.71
Easy Listening	2.97	1.98	2.97		1.98	9.90
Electronica/Dance	1.98	3.96	1.98	2.97	0.99	11.88
Heavy Metal	3.96	2.97		2.97	1.98	11.88
Hip Hop/Rap	1.98	2.97	3.96	3.96	3.96	16.83
Jazz	19.80	15.84	15.84	5.94	21.78	79.20
Latin	82.17	77.22	80.19	63.36	79.20	382.14
Metal	61.38	53.46	25.74	65.34	55.44	261.36
Pop	0.99	8.91	9.90	7.92		27.72
R&B/Soul	7.92	8.91	6.93	9.90	6.93	40.59
Reggae	5.94	6.93	7.92	5.94	2.97	29.70
Rock	178.20	155.43	156.42	162.36	174.24	826.65
Rock And Roll	0.99	1.98	0.99	1.98		5.94
Sci Fi & Fantasy		9.95	17.91	11.94		39.80
Science Fiction		3.98	3.98	1.99	1.99	11.94
Soundtrack	3.96	3.96	4.95	3.96	2.97	19.80
TV Shows		25.87	27.86	25.87	13.93	93.53
World	2.97	2.97	0.99	2.97	2.97	12.87
<b>Total</b>	<b>449.46</b>	<b>481.45</b>	<b>469.58</b>	<b>477.53</b>	<b>450.58</b>	<b>2,328.60</b>

# Goals of DWH

- Reporting → Dashboard



# Goals of DWH

- **Data mining** is used to:
  - Perform what-if analyses
  - Perform predictions ("predictive analysis")
  - Facilitate the decision process
- Data mining applications use sophisticated statistical and mathematical techniques
- Reports are less critical

# Advantages of DWH

- **High ROI (return on investment)**
  - Large investments with potentials high ROI after a short period.
- **Competitive advantage**
  - Decision makers get access to data that was not available, unknown or unused before.

# Advantages of DWH

- **Increased productivity of corporate decision-makers**
  - Decision makers get one consistent view on the enterprise
    - Because data of different sources can be integrated to one consistent view, that is subject oriented and keeps history.
  - Decision makers can make more substantial, more accurate and more consistent analysis
    - Tools can help turn data into useful information.

# Problems associated with DWH

- **Underestimation of resources (costs) for ETL**
  - Extraction, transformation and loading of data in the DWH takes a major part of the development time
  - Projects might take years
- **Hidden problems with source systems**
  - Are sometimes only discovered after years
  - Can be solved either in DWH or in operational DB
  - E.g. fields allowing NULL values: some offices never fill out the fields, although data is available and can be useful

# Problems associated with DWH

- **Required data not captured**
  - Change actual system or develop separate system for those data
  - E.g. the data a customer has registered is not captured
- **Increased end-user demands**
  - Demand for more user friendly, powerful and sophisticated tools
  - Enhanced load of IT personnel
  - Better end user training



# Problems associated with DWH

- **Data homogenization**
  - Trying to focus on similarities between data might lower the use of the data
  - E.g. similarities between sale and rent of properties
- **Need for concurrent support of several (historical) versions**
  - Operational systems evolve (i.e. database schema changes over time) but data from older versions and newer versions resides together in the DWH
  - Might be challenging

# Problems associated with DWH

- **High demand for resources**
  - E.g. disk space
- **Data ownership**
  - Sensitive departmental data (e.g. HR) is highly secured in HR department but is widely available in DWH
- **High maintenance**
  - Each change in business processes or in sources systems influences the DWH (both structure and ETL)

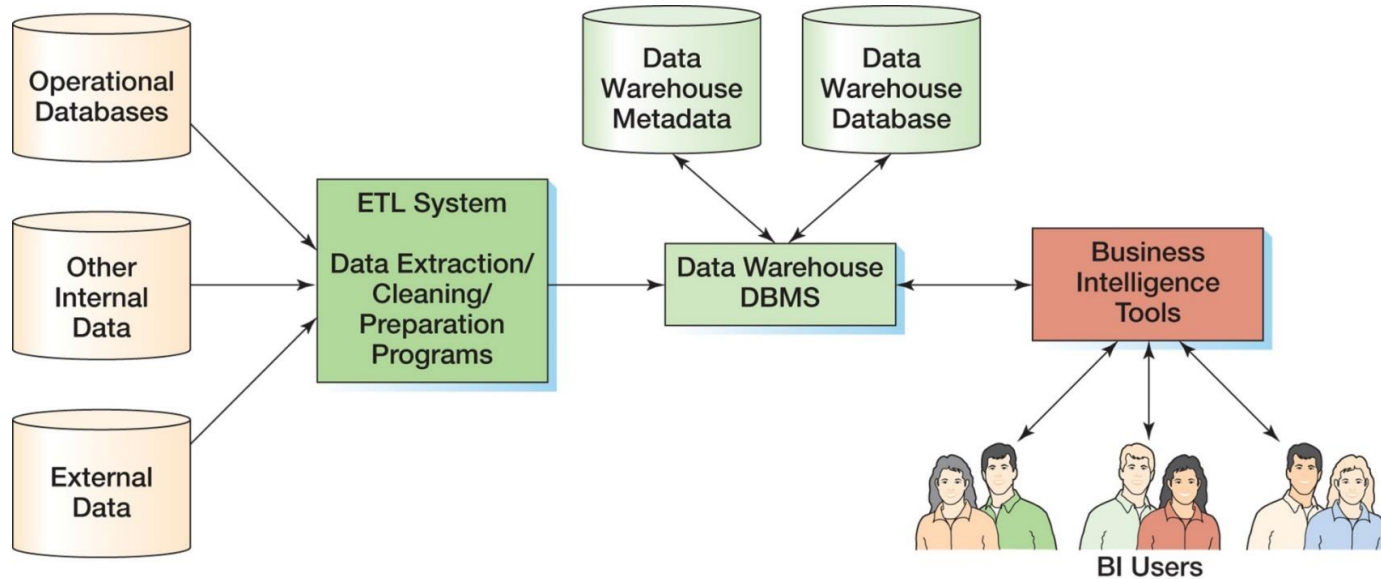
# Problems associated with DWH

- **Long projects**
  - Development can take years
  - Phased development through data marts (see further) might be recommended
- **DWH creates expectation of user ‘empowering’:**
  - Make own reports, analyses
  - Less need for IT
  - But:
    - Meta dictionary that describes data in DWH is necessary
    - Dependency of a few specialists remains

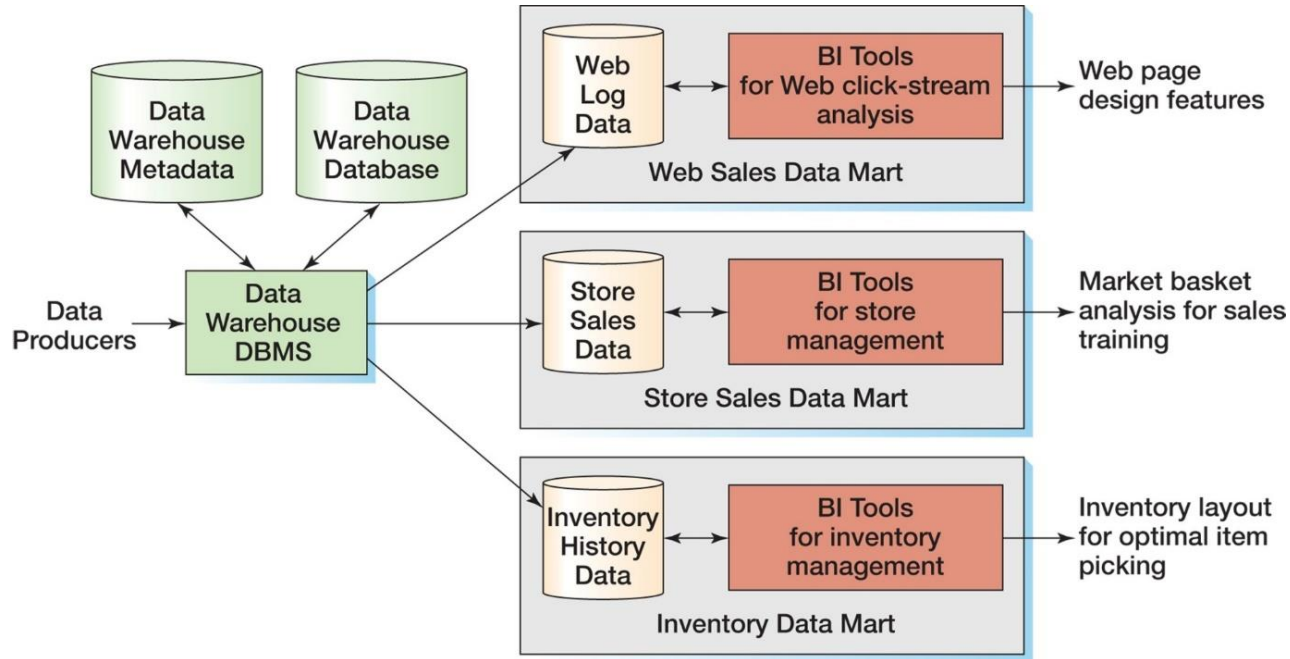
# Problems associated with DWH

- **Complexity of integration**
  - Different DWH tools have to work together smoothly
- **Complex change and version management**
  - Consistency in reporting between versions of underlying databases
  - **Night might be too short for ETL**

# DWH components



# DWH & Data marts



# Datamart

A DB existing of a **subset of company data** to support the needs of a particular business unit for data analysis, or, to support users sharing the same needs for analyzing **business processes**.

- Why a data mart?
  - To give users access to the data they analyse most frequently
  - To offer data in a way that corresponds to the collective view of a group of users in a department or a group of users in the same business process
  - To improve response time by offering lower data volumes
  - To offer data in a format that fits the tools used by end users (OLAP, datamining tools)
  - Reduction of complexity in the ETL process
  - Reduction of cost as opposed to the setup of a complete DWH



# Design

# Design of a DWH

- What is a datamart?

A datamart is a subset of the database created for a specific purpose. Usually, that purpose is to support a particular department.

- 2 development methodologies
  - Kimball
  - Inmon


# Design of a DWH

- **Inmon**

- Creation of a data model based on all **data** of the organisation;
- The relationships between the data are important.  
E.g. an address consists of a street name, a house number possibly with an addition, a postcode and a place name. No matter how you work or what business you are in, the above is always true. An address is an address.
- Because these kinds of connections are immutable, these connections should be the starting point for the design of the data warehouse.

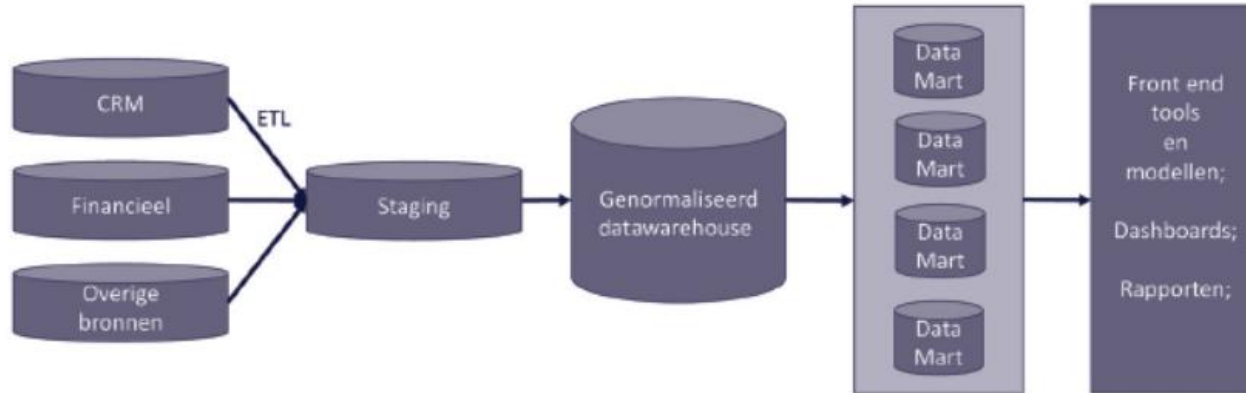
# Design of a DWH

- **Inmon**

- An Inmon data warehouse is a normalised database.
- It is modelled based on relationships between data that are stable over time (  an on changeable processes (Kimball → see further))
- An Inmon DWH is used to fill datamarts, it isn't used to make reports or do analyses;
- Enterprise Data Warehouse (EDW);

# Design of a DWH

- Inmon



# Design of a DWH

- **Kimball**

- Starts by identifying the information requirements (referred to as analytical themes) and associated **business processes** of the enterprise
  - Data Warehouse Bus Matrix;
- This first data mart is critical in setting the scene for the later integration of other data marts as they come online;
- The integration of data marts ultimately leads to the development of an EDW;
- Uses dimensionality modelling to establish the data model (called star schema) for each data mart.

# Kimball's Business Dimensional Lifecycle

- Guiding principle
  - Meet the information requirements of the enterprise by building
    - single,
    - integrated,
    - easy-to-use,
    - high-performance information infrastructure, which is delivered in meaningful increments of 6 to 12 months timeframes.

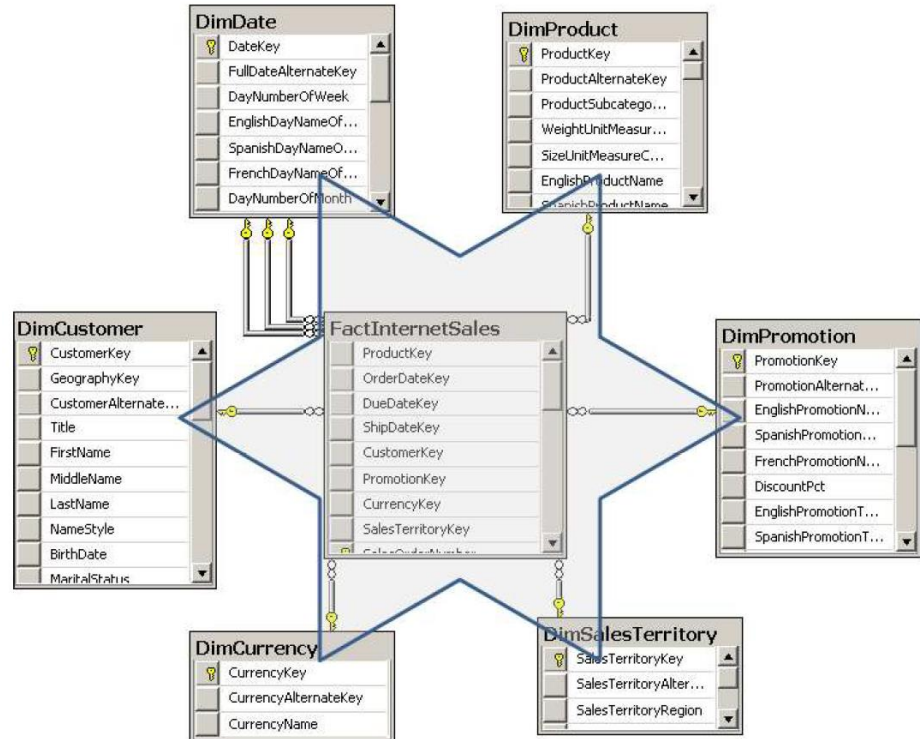
# Kimball's Business Dimensional Lifecycle

- Goal
  - Deliver the entire solution including
    - the data warehouse,
    - *ad hoc* query tools,
    - reporting applications,
    - advanced analytics
    - all the necessary training and support for the users.



# Star schema

A Star schema is a logical structure that has a **fact table** (containing factual data) in the center, **surrounded by denormalized dimension tables** (containing reference data).



## Remark:

- Natural keys from the operational system are available but not as a key in the star schema
- Surrogate integer keys are used instead because they are simpler and faster
- This way independence between OLTP and DWH is ensured

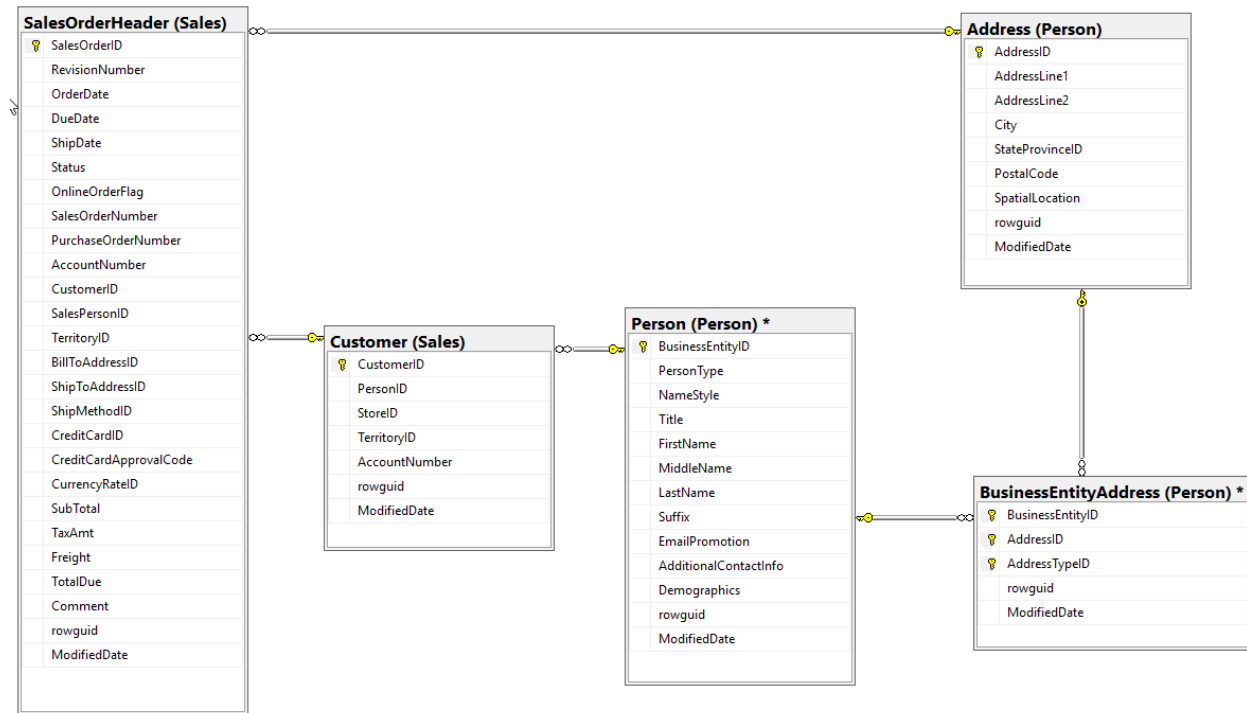
# Star schema

- **Fact table** contains data about facts
  - E.g. factual data about sales of property: sales price, commission percentage, margin, number of units sold, ...
- **Dimension table** contains reference information
  - E.g. property data (address, etc), buyer, owner, ...
- Facts are generated by events that happened (e.g. a sale)
- Most probably facts never change

# Dimensionality modelling: example

## ERD AdventureWorks

Often data in an OLTP system is highly normalized, which causes you might need all these tables for a simple sales report:



HO  
GENT


# The fact table


- Bulk of data in the data warehouse resides in fact tables, which can be extremely large.
- Important to treat fact data as read-only reference data that will not change.
- Most useful fact tables contain one or more numerical measures, or **'facts'** that occur for each record and are numeric and additive.

FactInternetSales	
	ProductKey
	OrderDateKey
3	DueDateKey
	ShipDateKey
3	CustomerKey
	PromotionKey
3	CurrencyKey
	SalesTerritoryKey
	🔑 SalesOrderNumber
	🔑 SalesOrderLineNumber
	RevisionNumber
	OrderQuantity
	UnitPrice
3	ExtendedAmount
	UnitPriceDiscountPct
	DiscountAmount
	ProductStandardCost
	TotalProductCost
	SalesAmount
	TaxAmt
	Freight
	CarrierTrackingNumber
	CustomerPONumber
	OrderDate
	DueDate
	ShipDate

# The dimension tables

- Dimension tables usually contain descriptive textual information.
- Dimension attributes are used as the constraints in data warehouse queries.  
(e.g. in where or having clauses).
- Star schemas can be used to speed up query performance by denormalising reference information into a single dimension table.

DimDate	
	DateKey
	FullDateAlternateKey
	DayNumberOfWeek
	EnglishDayNameOfWeek
	SpanishDayNameOfWeek
	FrenchDayNameOfWeek
	DayNumberOfMonth
	DayNumberOfYear
	WeekNumberOfYear
	EnglishMonthName
	SpanishMonthName
	FrenchMonthName
	MonthNumberOfYear
	CalendarQuarter
	CalendarYear
	CalendarSemester
	FiscalQuarter
	FiscalYear
	FiscalSemester

DimSalesTerritory	
	SalesTerritoryKey
	SalesTerritoryAlternateKey
	SalesTerritoryRegion
	SalesTerritoryCountry
	SalesTerritoryGroup
	SalesTerritoryImage

# Snowflake schema

Snowflake schema is a variant of the star schema that has a **fact table in the centre**, surrounded by **normalised dimension tables**.

DimDate	
DateKey	
FullDateAlternateKey	
DayNumberOfWeek	
EnglishDayNameOfWeek	
SpanishDayNameOfWeek	
FrenchDayNameOfWeek	
DayNumberOfMonth	
DayNumberOfYear	
WeekNumberOfYear	
EnglishMonthName	
SpanishMonthName	
FrenchMonthName	
MonthNumberOfYear	
CalendarQuarter	
CalendarYear	
CalendarSemester	
FiscalQuarter	
FiscalYear	
FiscalSemester	

FIGURE 1-4 The *DimDate* denormalized dimension.

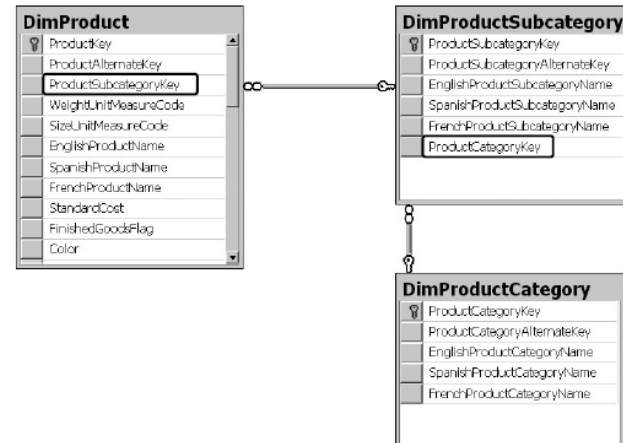


FIGURE 1-5 The *DimProduct* normalized dimension.

# Specific Schema Issues

- Surrogate keys
- Granularity of the fact table
- Factless Fact Tables
- Optimizing the dimension tables
- Defining junk dimensions
- Defining outrigger tables
- Slowly changing dimensions
- Rapidly changing dimensions

# Surrogate keys

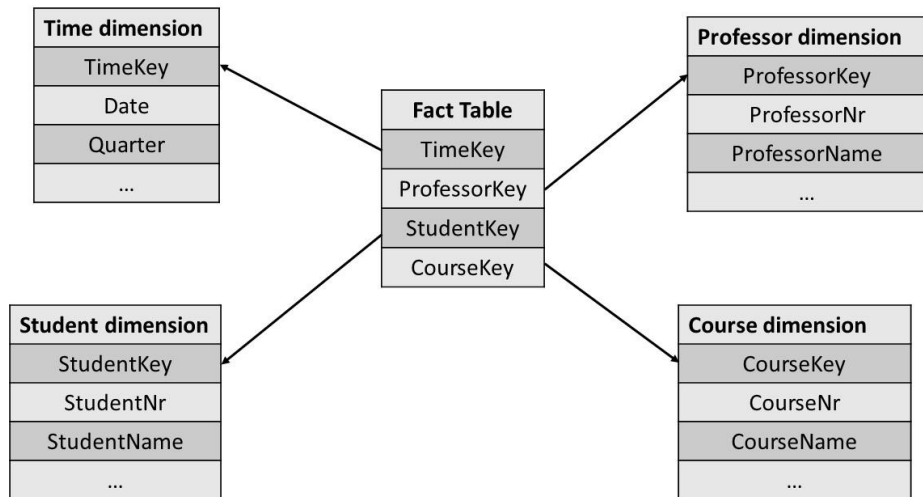
- Meaningless integers
  - StoreKey, ProductKey, ShipperKey, ...
- Cannot use business keys because of their functional meaning (= can be changed in case of a merger)
- Surrogate keys essentially buffer the data warehouse from the operational environment
- Business keys are usually bigger in size
- Business keys are also often re-used over longer periods of time



# Granularity of the fact table

- Level of detail of one row of the fact table
- Higher (lower) granularity implies more (fewer) rows and more (fewer) questions that can be answered
- Trade-off between level of detailed analysis and storage requirements
- Examples:
  - 1 tuple of the fact table corresponds to 1 line on a purchase order
  - One tuple of the fact table corresponds to one purchase order
  - One tuple of the fact table corresponds to all purchase orders made by a customer

# Factless Fact Tables



This data warehouse design allows you to answer questions such as:

- Which professor teaches the highest number of courses?
- What is the average number of students that attend a course?
- Which course has the maximum number of students?

# Optimizing the dimension tables

- Dimension tables should be heavily indexed to improve query execution time
- On average between 5 and 10 indexes
- E.g. DateDimension
  - TimeKey, Date, DayOfWeek, DayOfMonth, DayOfYear, Month, MonthName, Year, LastDayInWeekFlag, LastDayInMonthFlag, FiscalWeek, HolidayFlag, ...

	DateKey	FullDateAlterna...	DayNumberOf...	EnglishDayNam...	DayNumberOf...	DayNumberOf...	WeekNumberO...	EnglishMonthN...	MonthNumberO...	CalendarQuarter	Calendar...
►	20050101	2005-01-01	7	Saturday	1	1	1	January	1	1	2005
	20050102	2005-01-02	1	Sunday	2	2	2	January	1	1	2005
	20050103	2005-01-03	2	Monday	3	3	2	January	1	1	2005
	20050104	2005-01-04	3	Tuesday	4	4	2	January	1	1	2005
	20050105	2005-01-05	4	Wednesday	5	5	2	January	1	1	2005
	20050106	2005-01-06	5	Thursday	6	6	2	January	1	1	2005

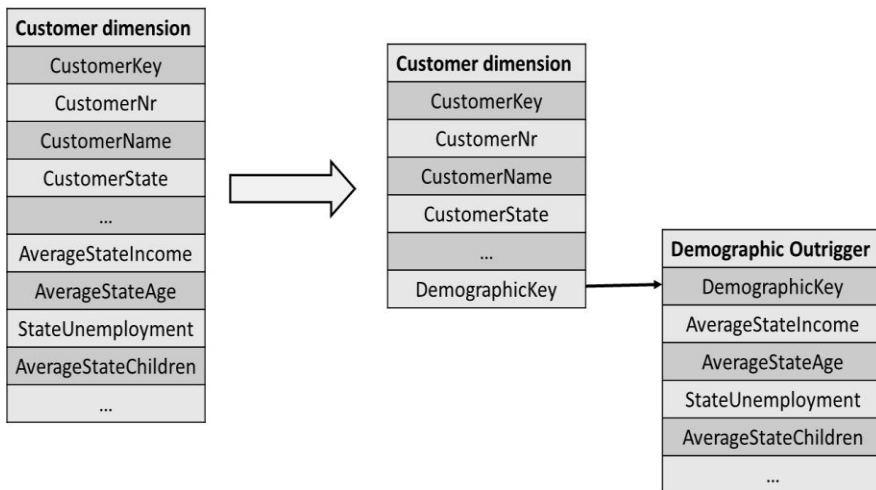
# Junk Dimensions

- Deal with low cardinality attribute types such as flags or indicators
- Example: On-line Purchase (Y/N), Payment (cash or credit-card), Discount (Y/N)
- Junk dimension is a dimension that simply enumerates all feasible combinations of values of the low cardinality attribute types
- Advantage?

JunkKey1	On-line purchase	Payment	Discount
1	Yes	Credit-card	Yes
2	Yes	Credit-card	No
3	No	Credit-card	Yes
4	No	Credit-card	No
5	No	Cash	Yes
6	No	Cash	No

# Outtrigger tables

- Store a set of attribute types of a dimension table which are highly correlated, low in cardinality and updated simultaneously



# Slowly Changing Dimensions

- Dimensions that change slowly and irregularly over a period of time
- Example: customer segment ranging from AAA, AA, A, BBB, ... to C, determined on a yearly basis

# Slowly Changing Dimensions

- Approach 1

## OLD STATUS

CustomerKey	CustomerNr	CustomerName	Segment
123456	ABC123	Bart Baesens	AA

## NEW STATUS

CustomerKey	CustomerNr	CustomerName	Segment
123456	ABC123	Bart Baesens	AAA

→ Loss of previous state

# Slowly Changing Dimensions

- Approach 2

## OLD STATUS

CustomerKey	CustomerNr	CustomerName	Segment	Start_Date	End_Date	Current_Flag
123456	ABC123	Bart Baesens	AA	27-02-2014	31-12-9999	Y

## NEW STATUS

CustomerKey	CustomerNr	CustomerName	Segment	Start_Date	End_Date	Current_Flag
123456	ABC123	Bart Baesens	AA	27-02-2014	27-02-2015	N
123457	ABC123	Bart Baesens	AAA	28-02-2015	31-12-9999	Y

Every time an attribute changes, a completely new record is created. There are 'current' records and 'closed' records.



# Slowly Changing Dimensions

- **Approach 3**

## **OLD STATUS**

CustomerKey	CustomerNr	CustomerName	Segment
123456	ABC123	Bart Baesens	AA

## **NEW STATUS**

CustomerKey	CustomerNr	CustomerName	Old Segment	New Segment
123456	ABC123	Bart Baesens	AA	AAA

You keep both the current and previous value of an attribute in two different columns. That way, current and previous values can be easily compared (no older values available)

# Slowly Changing Dimensions

- Approach 4

Customer

CustomerKey	CustomerNr	CustomerName	Segment
123457	ABC123	Bart Baesens	AAA

Customer history

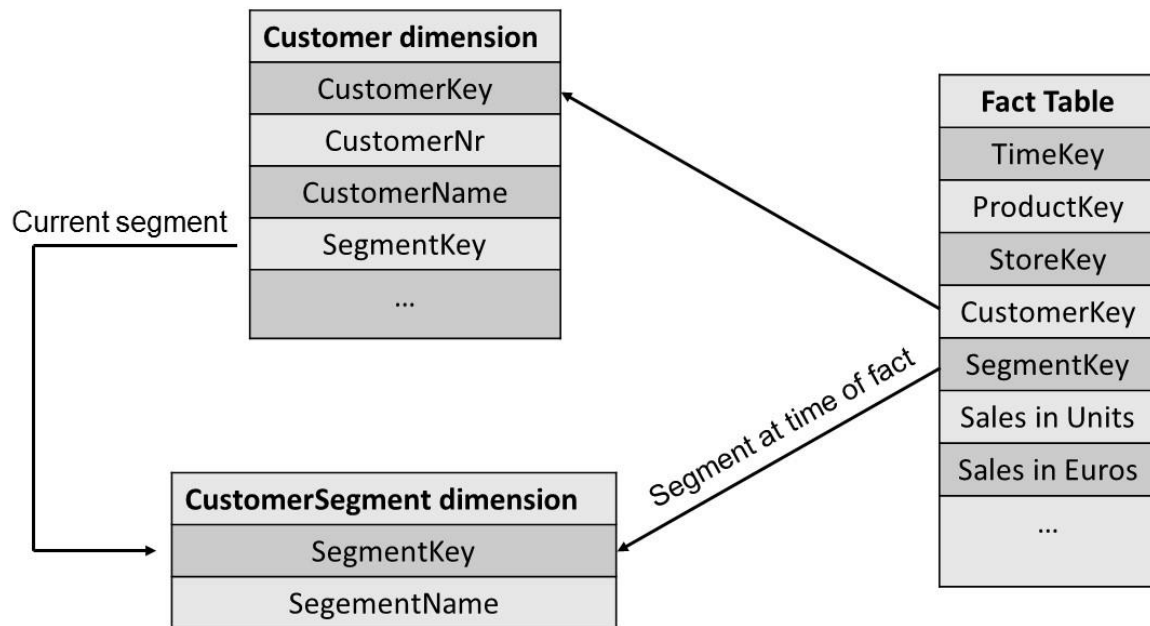
CustomerKey	CustomerNr	CustomerName	Segment	Start_Date	End_Date
123456	ABC123	Bart Baesens	AA	27-02-2014	27-02-2015
123457	ABC123	Bart Baesens	AAA	28-02-2015	31-12-9999

# Rapidly Changing Dimensions

- Dimensions that change rapidly and regularly over a period of time
- Example: customer segment ranging from AAA, AA, A, BBB, ... to C, determined on a daily basis
- Approaches 2 and 4 discussed in the previous section will result into a lot of rows
- Split customer information into stable (e.g., gender, marital status, ...) and rapidly changing information which is put in mini-dimension table

# Rapidly changing dimensions

## Approach 1:



# Rapidly changing dimensions

## Approach 1 with sample data:

### Fact table

TimeKey	ProductKey	StoreKey	CustomerKey	SegmentKey	Sales in Units	Sales in Euros
200	50	100	1200	1	6	167.94
210	25	130	1400	4	3	54
180	30	150	1000	3	12	384
...						

### Customer Dimension

CustomerKey	CustomerNr	CustName	SegmentKey
1000	20	Bart Baesens	2
1200	10	Wilfried Lemahieu	1
1400	5	Seppe vanden Broucke	1

Rapidly changing part

### CustomerSegment Dimension

SegmentKey	SegmentName
1	A
2	B
3	C
4	D

Stable part

# Advantages of the dimensional model

- Predictable and standard form of the underlying dimensional model offers important advantages:
  - **Efficiency**
    - A consistent DB structure allows tools to have efficient access to the data
  - **Ability to handle changing requirements**
  - The model can easily adapt to changing needs because each dimension is equivalent to the fact table
    - Ideal for ad hoc queries
  - **Extensibility**
    - Adding new facts
    - Adding new dimensions
    - Adding attributes to dimensions

# Advantages of the dimensional model

- Predictable and standard form of the underlying dimensional model offers important advantages:
  - **Ability to model common business situations**
  - **Predictable query processing**
    - The way tables are used is predictable (not the queries themselves)

# DM and ER models

- **Entity Relationship Diagrams**

- Used to design the DB of OLTP systems
- Basics: remove redundancies
  - Redundancy causes update/delete/insert anomalies
- Ad hoc queries are more difficult
  - Lots of tables can be involved: deep join constructs

- **Dimensional Modelling**

- Used for design of DWH or data mart
- Intuitive storage and high performance consulting of data



# DM and ER models

- A single ER model normally decomposes into multiple DMs.
- Multiple DMs are then associated through ‘shared’ dimension tables.

# Dimensional modelling Stage

- Design choices:
  - Choose granularity level:
    - Type 1: the number of dimensions determines the granularity level of the analysis
    - Type 2: every order, summarized by month, quarter, ...
  - Duration measures how far back in time the fact table goes
  - Slowly changing dimension problem means the proper description of the old dimension data must be used with the old fact data.
    - Type 1: the attribute that changes is **overwritten**
    - Type 2: if an attribute changes a **new record** is added to the dimension table
    - Type 3: make sure **old and new value** of the attribute is available in the same record