

Next Generation DWH Modeling

An overview of DWH modeling methods



Ronald Kunenborg | www.grundsatzlich-it.nl



Topics



- Where do we stand today



- Data storage and modeling through the ages



- Current data warehouse modeling methods



- Where next?



- Food for thought



Where do we stand today

Fun facts from a 2012 TDWI-survey by Kalido:

- ▶ 64% need more than 1 month to integrate new data or integrate changes
- ▶ 24% spend more than \$1,000,000 annually
- ▶ 31% employ more than 20 people to maintain the BI environment
- ▶ 85% agree IT can't keep up with business
- ▶ Only 56% agree that IT and business are aligned

We need to do better...

Where do we stand today

What do we need in order to be successful?

- ▶ Have a business-oriented view on the data
- ▶ Have good performance for our queries
- ▶ Integrate different sources with ease
- ▶ Be able to implement changes to reports quickly
- ▶ Be able to handle ‘business changing its mind’
- ▶ Handle large data volumes at low cost

“Deliver cheap reports in business terms, fast!”

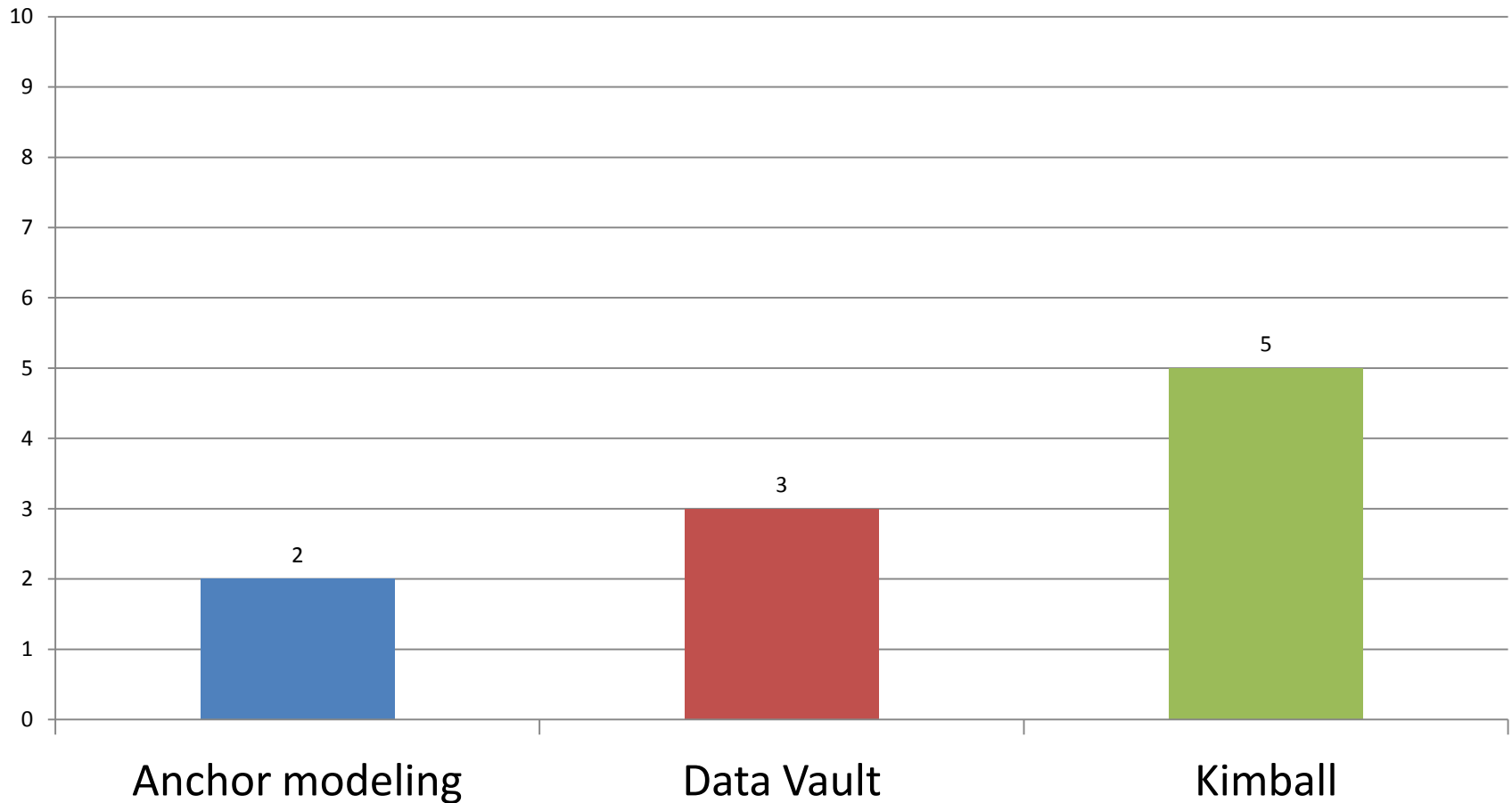
Our data models and practices need to support this.

How are we doing?



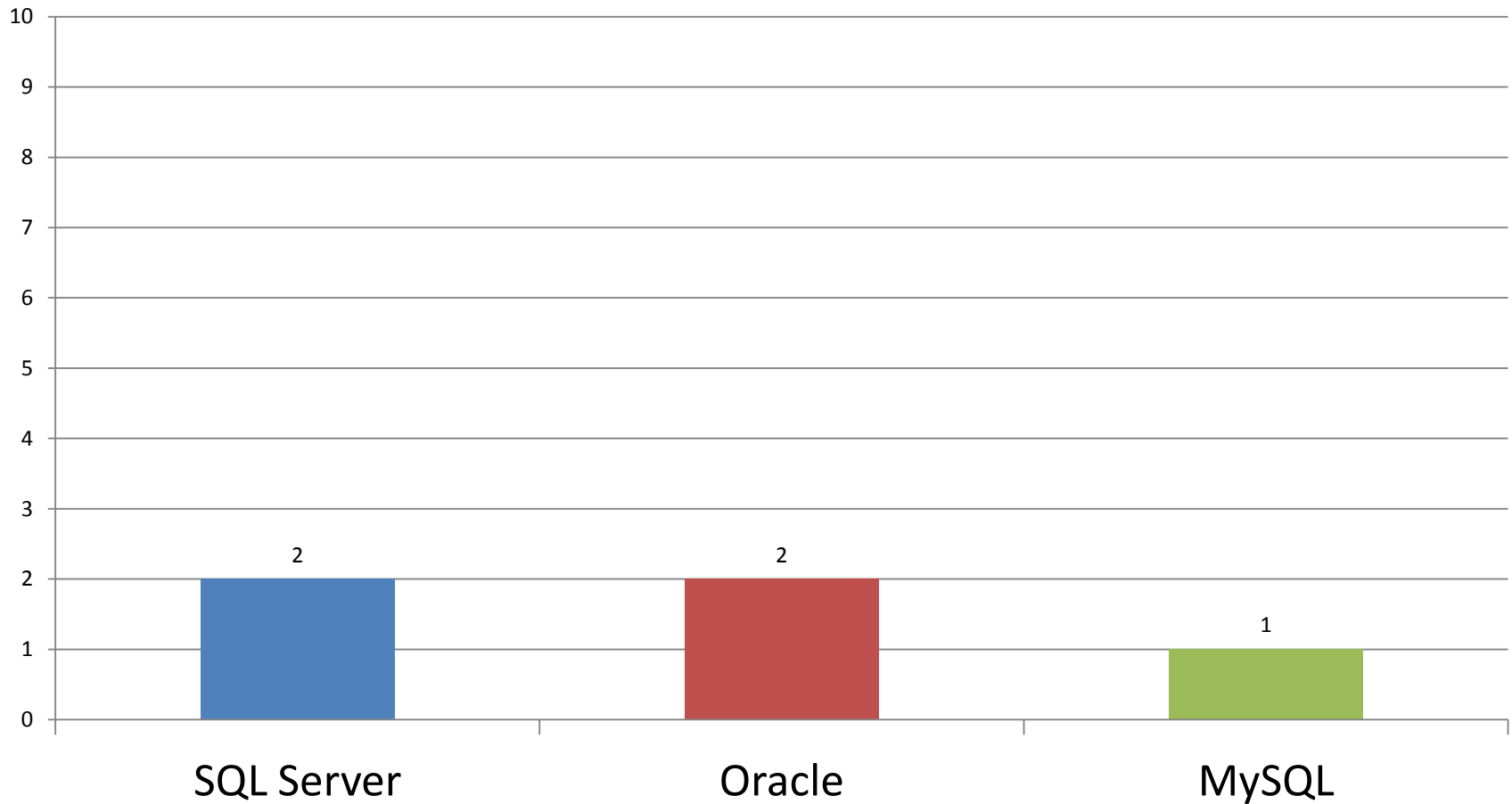
Where do we stand today

Modeling methods used



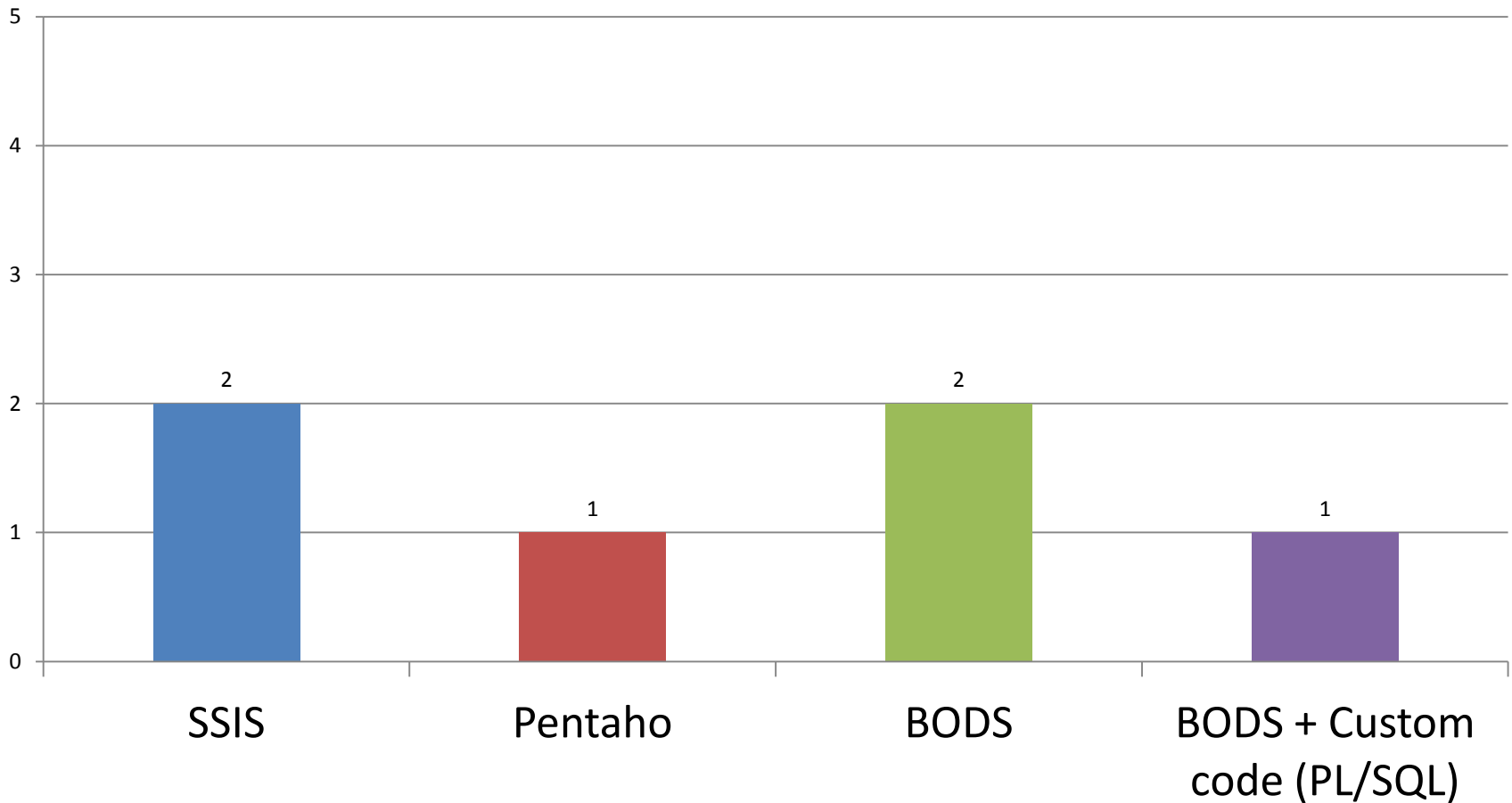
Where do we stand today

Databases used



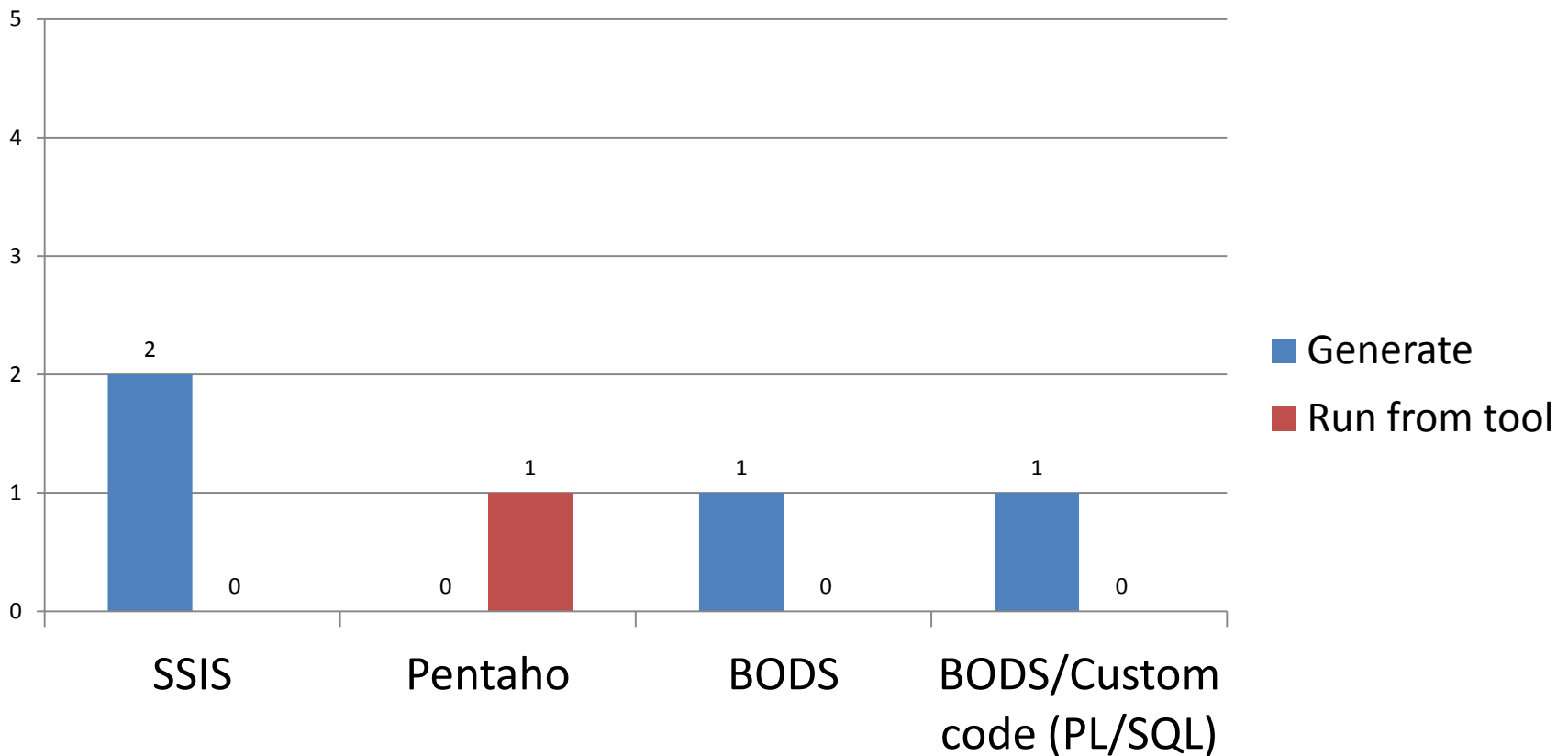
Where do we stand today

ETL-tools used



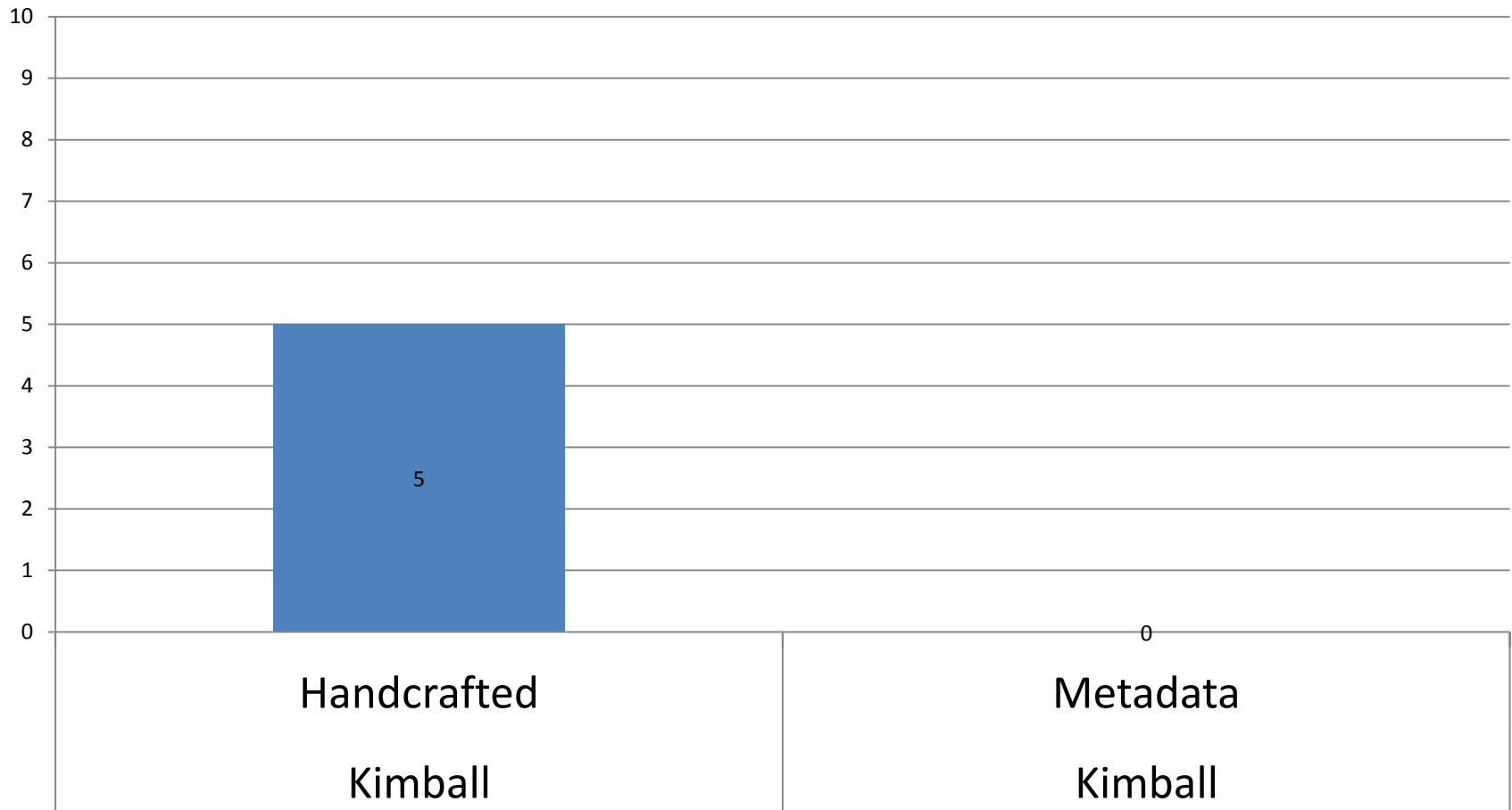
Where do we stand today

ETL: tool vs method (generate script or run from tool)



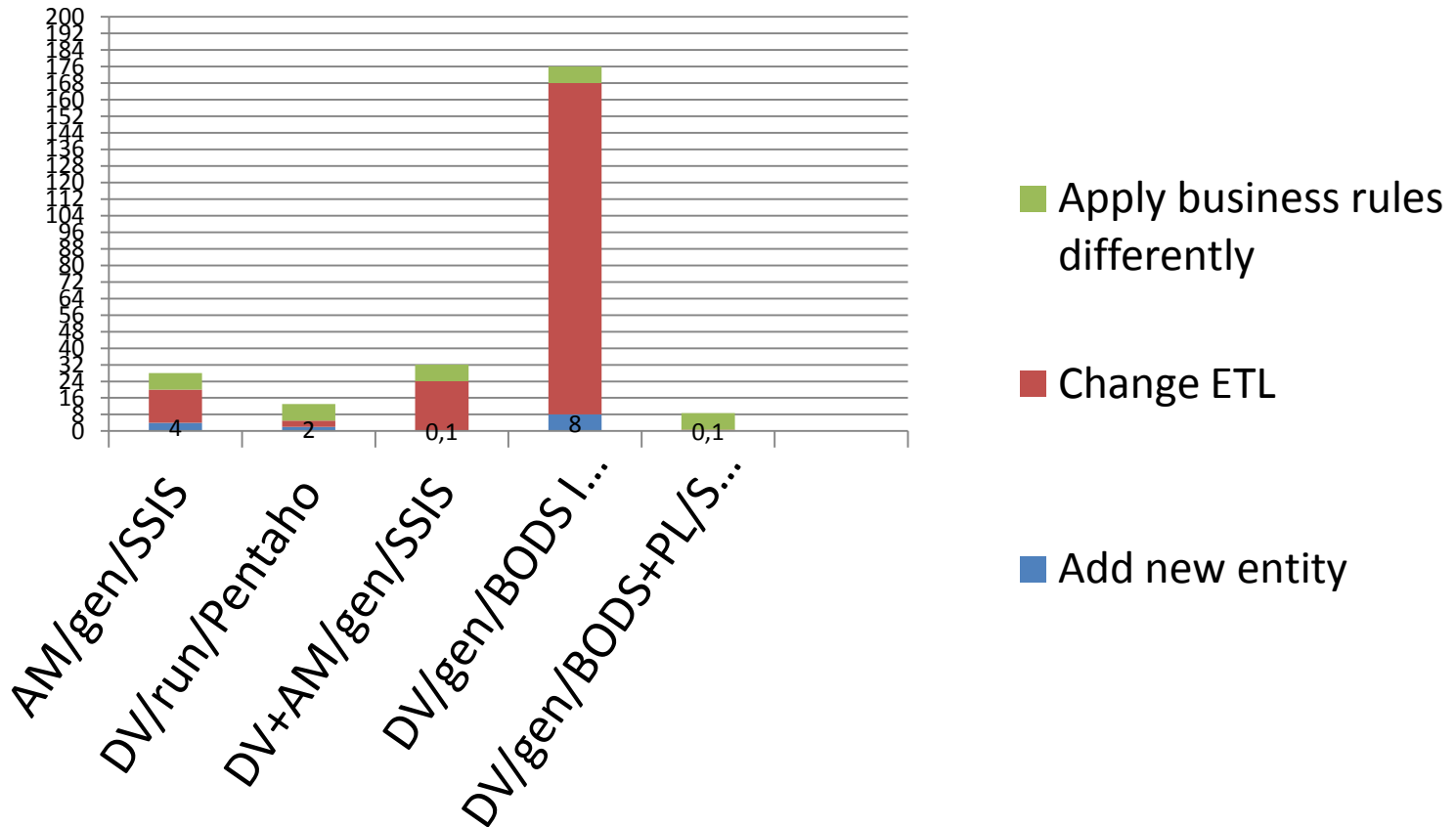
Where do we stand today

Building datamarts: manual or automated



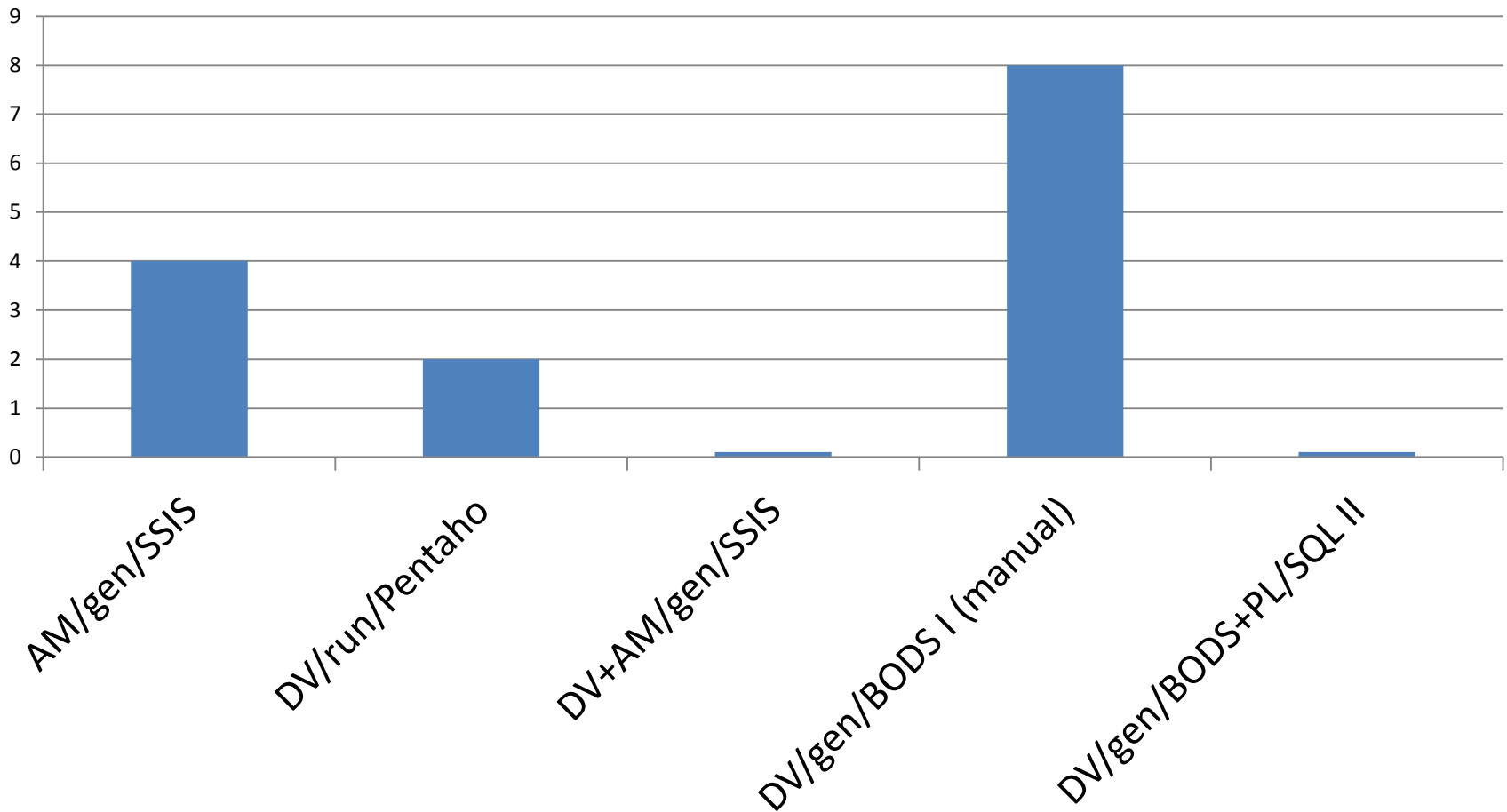
Where do we stand today

ETL: hours to add entity / rebuild ETL



Where do we stand today

Add new entity



Where do we stand today

So where does this leave us?

- ▶ We can automate the simple data logistics
- ▶ We can automate the historical storage area and save time on tedious ETL
- ▶ Building data marts is still a manual process
- ▶ (Complex) business rules are still hard

We satisfy some requirements.

Can we do better?



Where do we stand today



If we want to understand how we got here – we need to understand our past...

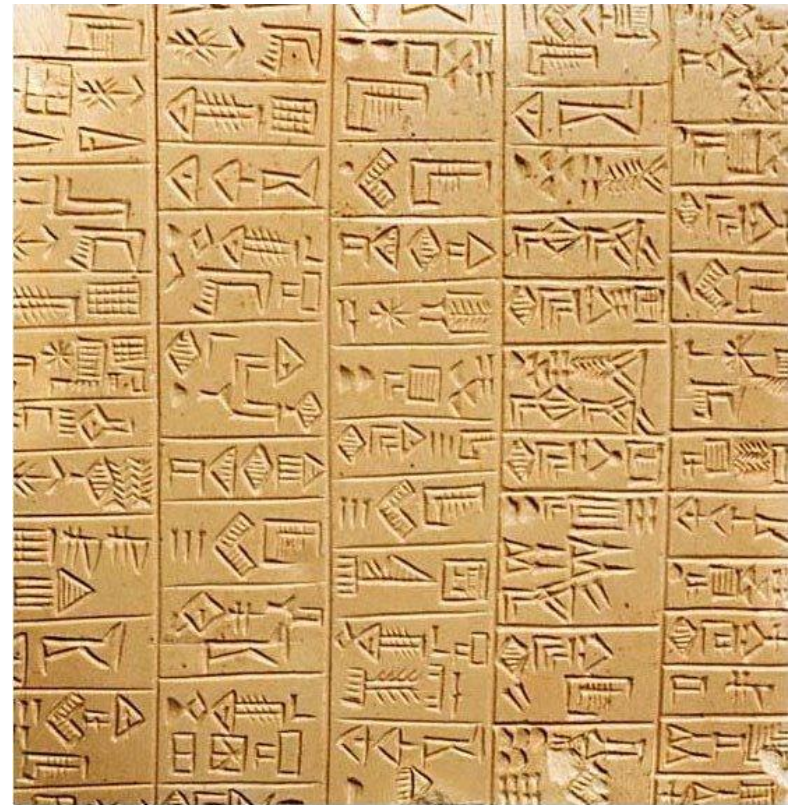
- ▶ Why did we start with data modeling in the first place?
- ▶ Pro's & cons of older methods
- ▶ Are data models really important?

Data storage through the ages

Sumeria, 3400 BC.

The dawn of civilization,
humanity grows and
starts to live in cities

Records needs to be kept
about food stores. It's no
longer possible to just
remember everything...



A list of temple gifts (Cuneiform)

This is a “big data” issue

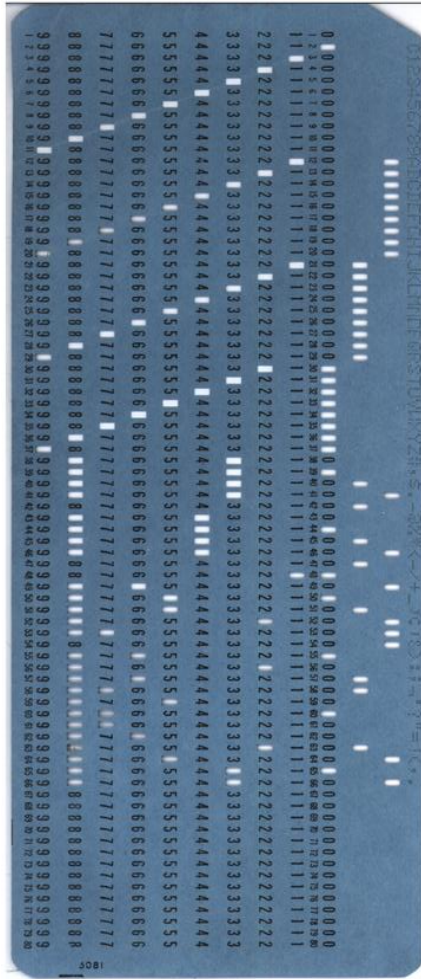
Data storage through the ages

Early capitalism sees a growing number of transactions for bankers: “big data” issues appear

In 1494, bookkeeping changes:

- ▶ Information about subject of transaction is separated from transaction details:
normalization appears
- ▶ Now we can change account information (balance) without changing transaction details

Data storage through the ages



- Punched cards appear around 1725
 - ▶ First used for information storage in 1832 as information grows
 - ▶ Expanded for the **1890 US census**
 - ▶ Enabled aggregations on a very large set of detail data for first time in history
 - ▶ Limited by machine speed
 - ▶ Early form of dimensional modeling

Data storage through the ages

After 1952, magnetic tape (electronic) starts to slowly replace punch cards (mechanical)

- ▶ Fast, huge storage (100 GB in 1974)
- ▶ Still only sequential
- ▶ Punch cards are completely gone by end of the 1970's



IBM 726 vacuum-column dual tapedrives

Data storage through the ages - conclusion

Hard disks are introduced around 1960 and change the entire field

- ▶ Enabled RANDOM data access and Online Transaction Processing (OLTP)
- ▶ Requirements differ from batch processing
- ▶ Databases & data modeling appear

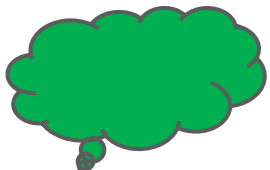


IBM 350 RAMAC (3,75 MB)

Data modeling through the ages

With random access, data is no longer “flat records”

- ▶ Data modelers organize the data on permanent storage to achieve the desired purposes
- ▶ Data models present the data and its relationships in a graphical format for better understanding
- ▶ What are our *logical* modeling “Lego”-blocks?



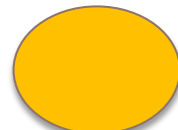
CONCEPT



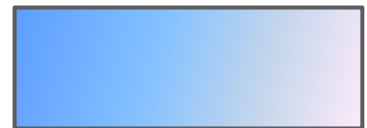
KEY



RELATIONSHIP



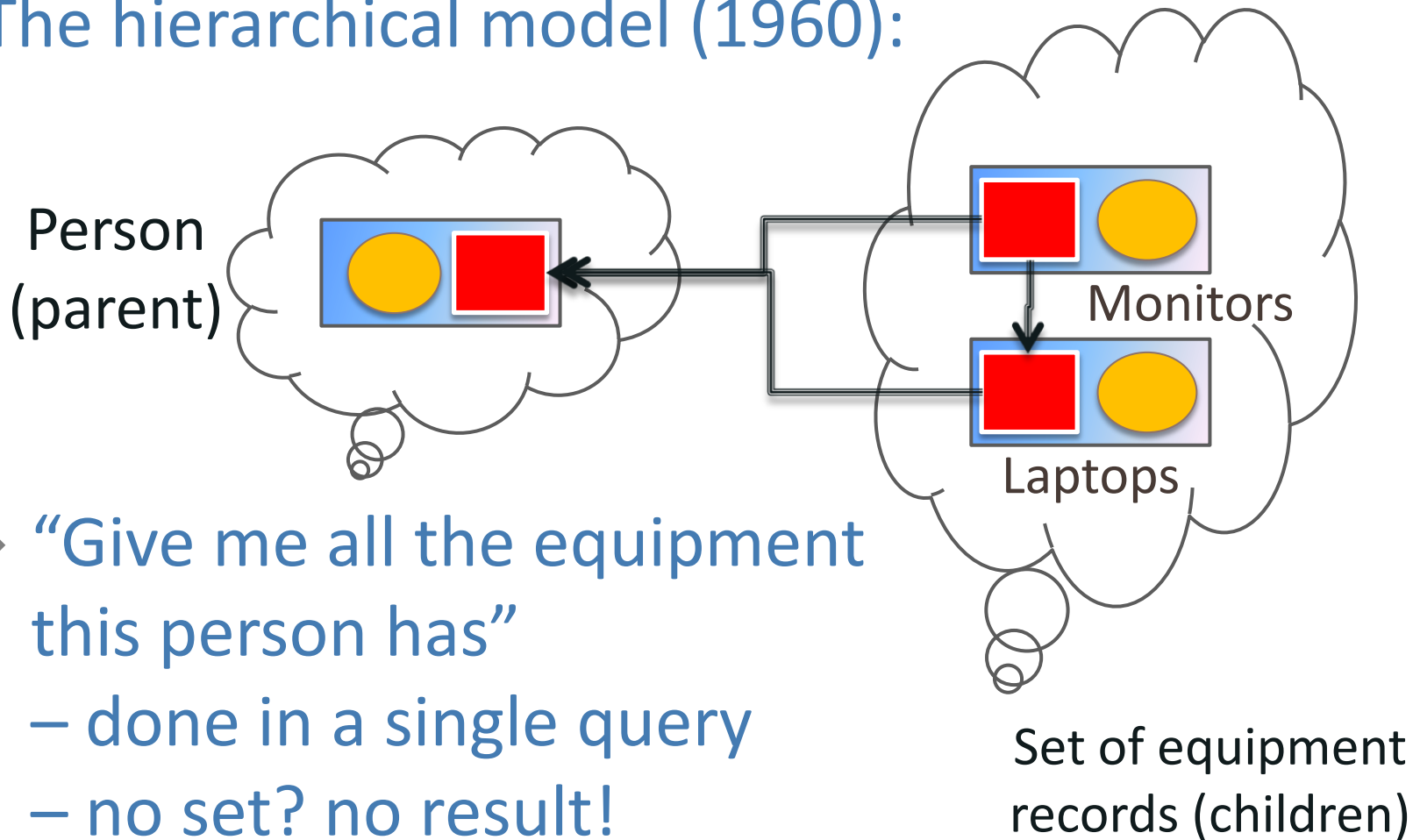
ATTRIBUTE



DATABASE
“TABLE”

Data modeling through the ages

The hierarchical model (1960):



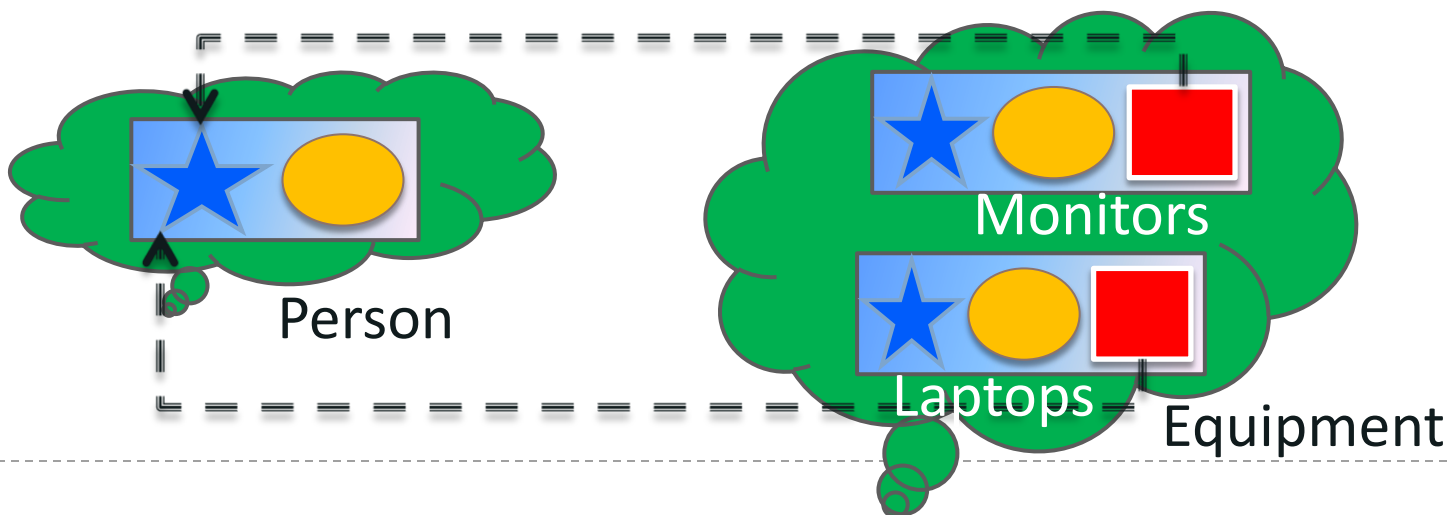
Data modeling through the ages

- ▶ Late 70's: Relational databases appear (Oracle)
- ▶ Data is stored as tables and their relationships
- ▶ Queries are declarative: say WHAT, not HOW
- ▶ Separating code and database makes building and maintaining software (much) cheaper
- ▶ The relational model replaces older database models by the end of the 80's
- ▶ Now we can do more queries with less people

Data modeling through the ages

Standard relational modeling focuses on speed and consistency of data through normalization

- ▶ This works well for transactional systems
- ▶ Easy to retrieve and update single items by key
- ▶ Queries over large datasets are slow



Data modeling through the ages

Decision support systems appear during the 80's.
The term “Data Warehouse” is coined by Bill Inmon.

- ▶ At first built using normalized data models

But:

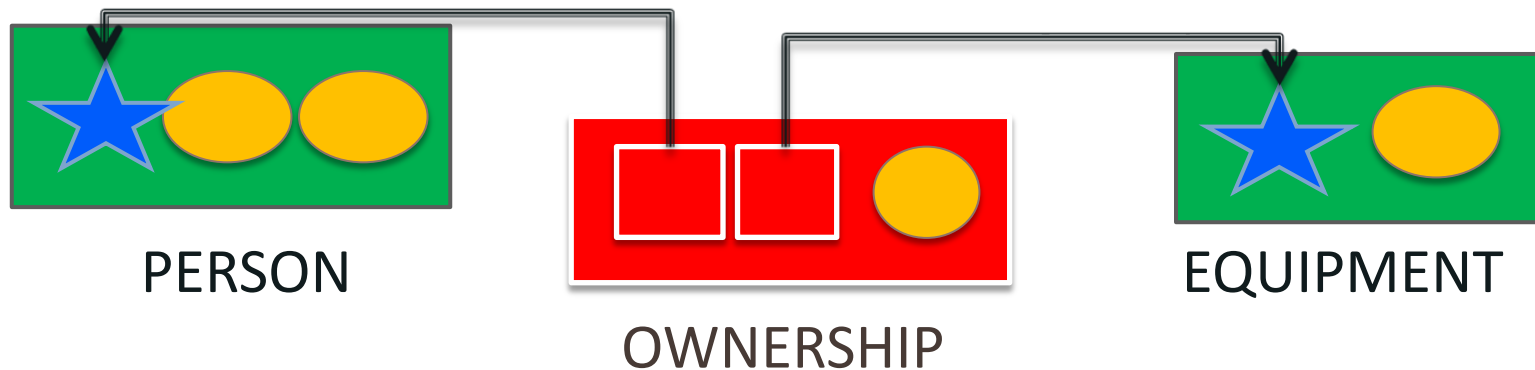
- ▶ Unpredictable queries on large datasets are slow
- ▶ Changes to source systems are hard to process and cascade into multiple DWH changes
- ▶ Integrating different source systems is quite difficult



Data modeling through the ages

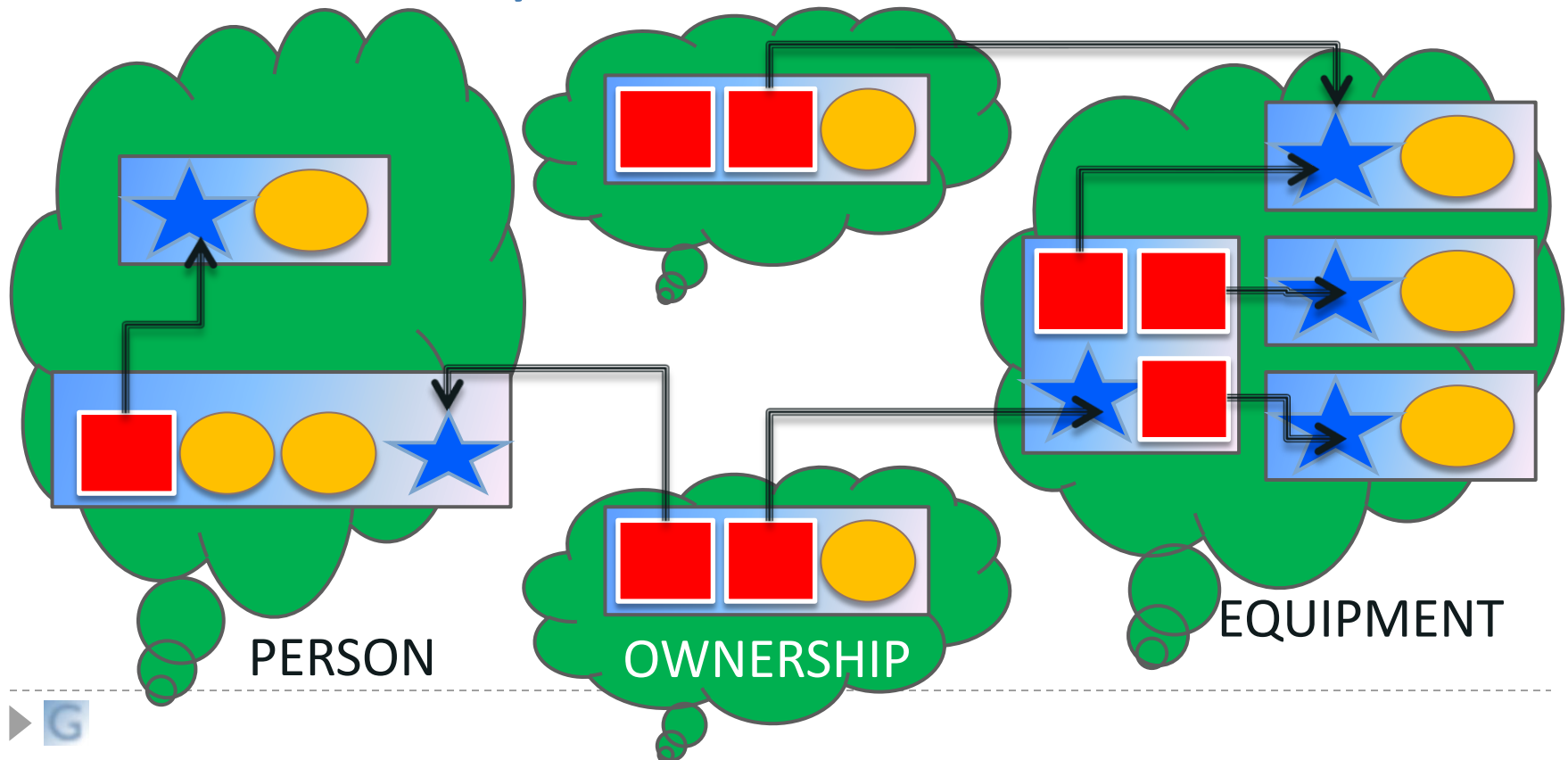
In the 90's we see the Dimensional model

- ▶ Concepts are stored in dimension-tables and relationships in fact-tables
- ▶ Easy to understand, build and query
- ▶ Less joins means faster queries



Data modeling through the ages

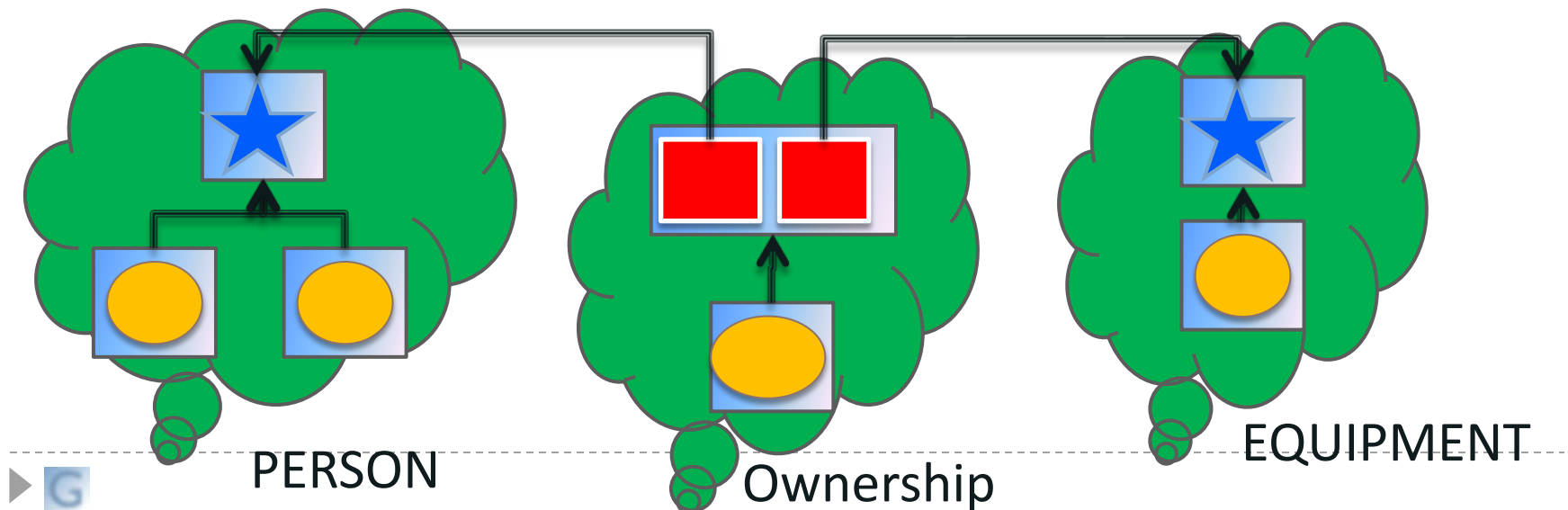
- ▶ Over the years the model degrades
- ▶ performance, understanding and maintainability suffer – often, a rebuild follows



Data modeling through the ages

In the early 00's Dan Linstedt publishes Data Vault:

- ▶ Business keys in Hubs
- ▶ business relationships in Links, (historical) attributes in Satellites
- ▶ Rules to maintain integrity of the data model



Data modeling through the ages

Data Vault advantages:

- ▶ **Resilient to change**
- ▶ Better load performance
- ▶ Easy to automate (except business rules)

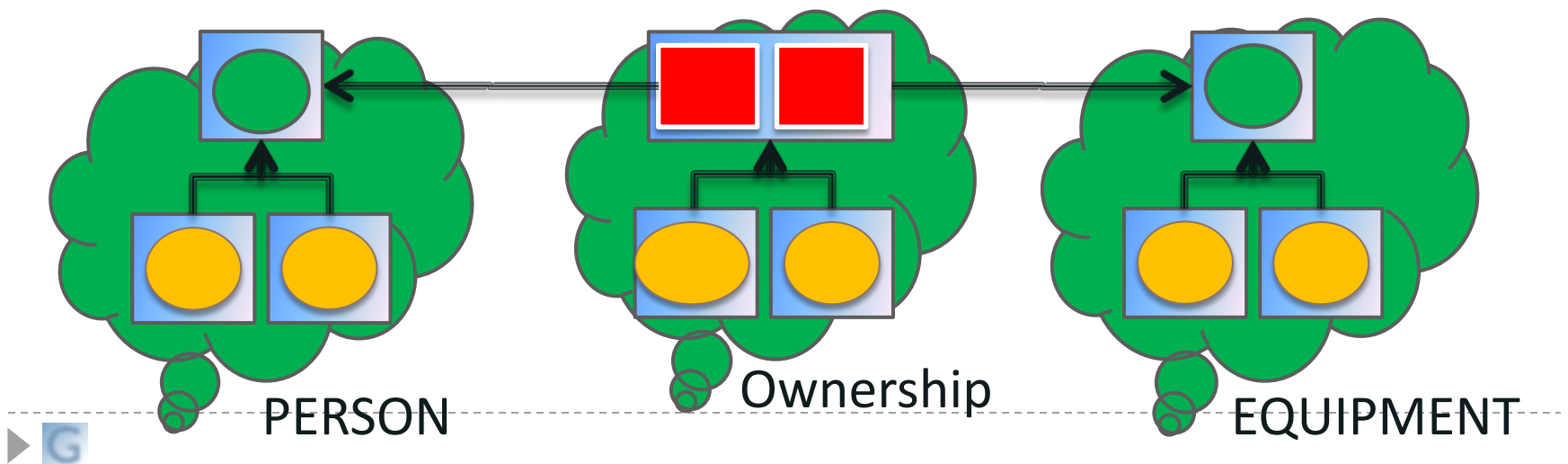
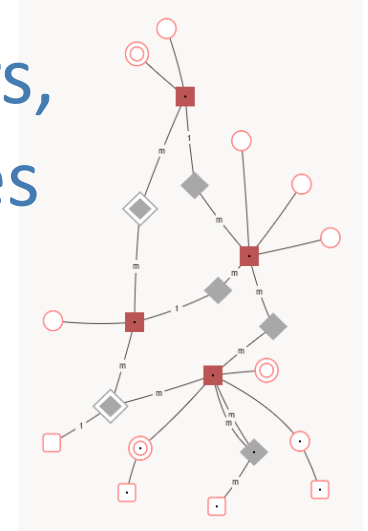
Data Vault disadvantages:

- ▶ Complex queries
- ▶ Lower query performance
- ▶ Misinterpretation of rules may be an issue

Data modeling through the ages

In the early 00's Anchor Modeling shows up too

- ▶ Concepts are defined around Anchors, relationships between Anchors in Ties
- ▶ Attributes are stored in Attributes
- ▶ Keys are just attributes



Data modeling through the ages

Anchor Model advantages:

- ▶ Minimal impact of changes in source systems
- ▶ A Kimball model is a set of views on this model
- ▶ Easy to automate (except business rules)

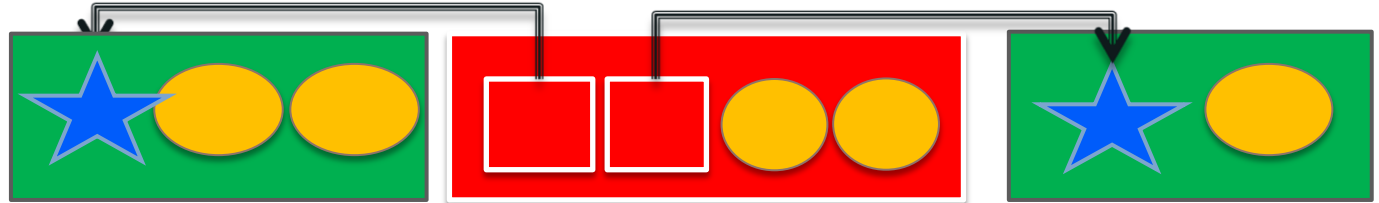
Anchor Model disadvantages:

- ▶ Query performance relies on the number of attributes in the query and on the specific abilities of the underlying RDBMS
- ▶ No rules on storing interpreted data

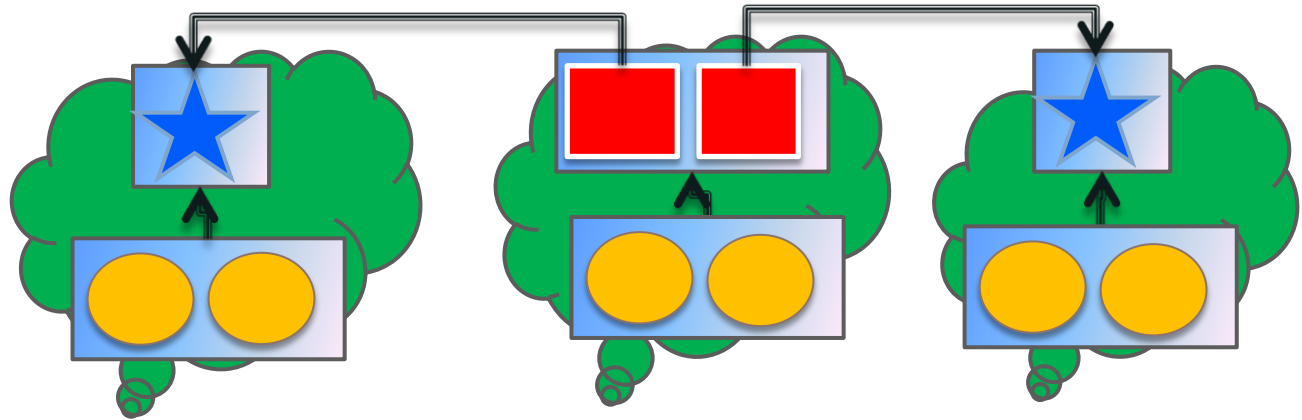


The different models

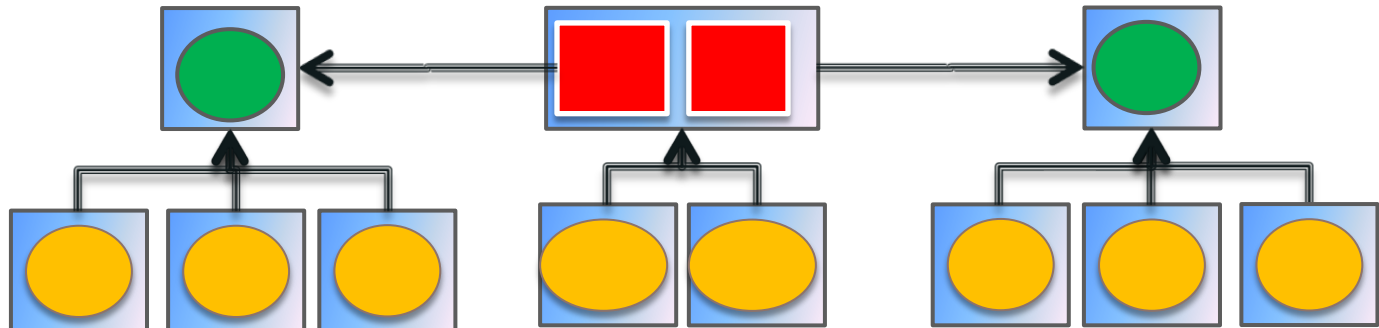
Kimball



Data
Vault



Anchor
model









The different models

The (very rough) differences:

- ▶ Kimball focuses on business concepts and ease of use
- ▶ Data Vault focuses on resilient long term storage, auditing and business administration
- ▶ Anchor Modeling focuses on business concepts and resilient long term storage
- ▶ 3NF (“Inmon”) focuses on historical replication of the source

Data modeling through the ages

So let's get back to our requirements:

- ▶ Have a business-oriented view on the data 
- ▶ Have good performance for our queries 
- ▶ Integrate different sources with ease 
- ▶ Be able to handle 'business changing its mind' 
- ▶ Implement changes to reports quickly 
- ▶ Handle large data volumes at low cost 

How can we change reports quickly?

What if we have terabytes of data? Or petabytes?

Where next?



Can we achieve our objectives using cheap(er)
“NoSQL” data stores?

- ▶ Best known alternative: Hadoop
- ▶ Hadoop is a replicated data store for unstructured data, with many servers holding parts of the data
- ▶ Structure is imposed by the (programmed) query
- ▶ SQL front-end (Hive) and distributed database (Cassandra) are available on top of the framework
- ▶ Basically, Hadoop is a huge number of “very fast clay tablets”



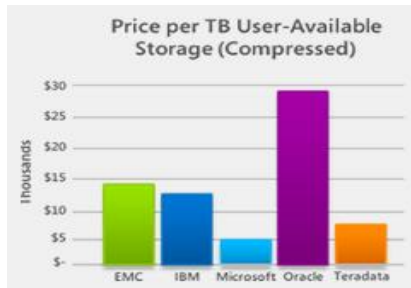
How does it stack up?

Our requirement was cheap storage or fast changes:

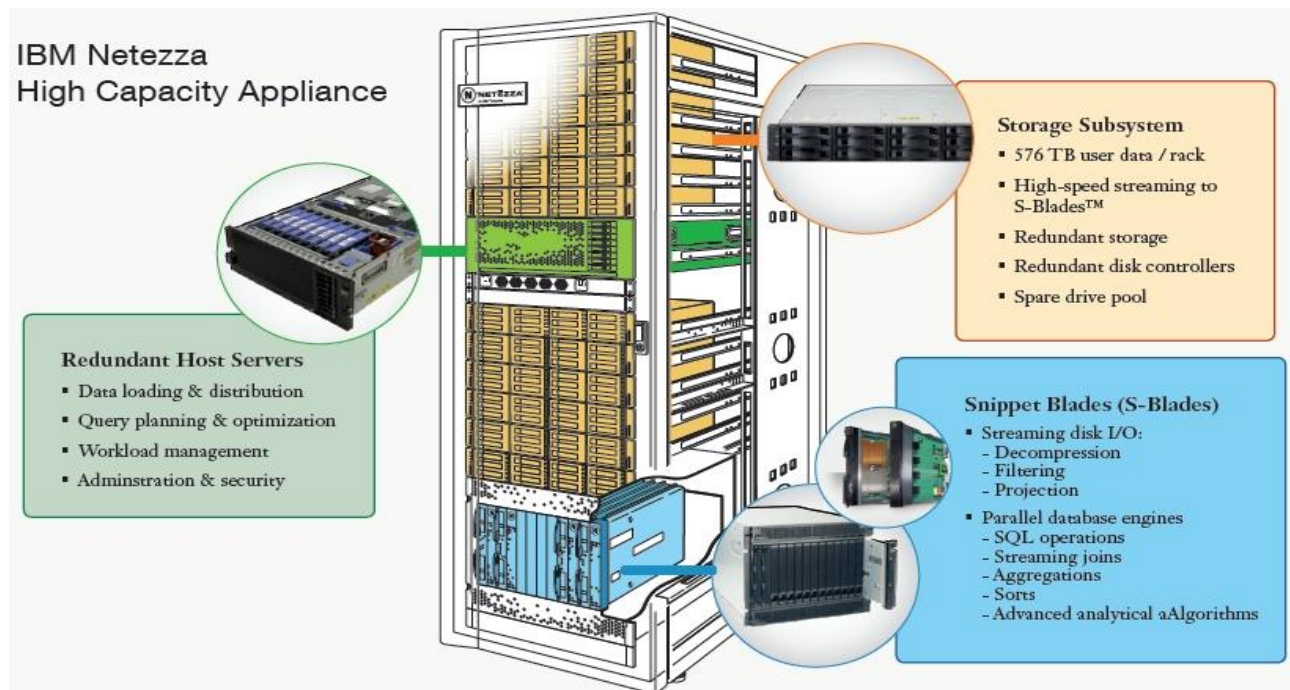
- ▶ For terabyte-sized volumes of data, Hadoop is cheaper in hardware and software cost than a commercial RDBMS or DWH appliance
- ▶ Two thirds of the cost of Hadoop lies in programming the queries and maintaining the system – Hadoop may have a very expensive TCO
- ▶ Facebook: “Hadoop is superior at exploring vast amounts of data with complete flexibility, but Relational is superior at traditional business questions” (TDWI, May 2013)

Where next?

Or: use a DWH appliance and trade hardware cost for lower labor and storage costs



Source: microsoft.com



http://www.theregister.co.uk/2011/06/23/ibm_netezza_high_capacity/

Competition: **TERADATA**  **GREENPLUM**  

Food for thought

Data modeling is very relevant,
but:

- ▶ Method trumps model
- ▶ Automation may make the choice of data model moot
- ▶ Automation of business rules beats automating simple ETL
- ▶ Our job is not just to model, but to make the trade-offs between different data models



An overview of modeling methods

► About the author:



Ronald Kunenborg obtained a Masters degree in Computer Science in 1994, before working as Scientific Programmer and Business Engineer at Mercedes-Benz for a decade.

During that time he was active in all the fields in computing, including building the Logistics and Marketing data warehouse and all the reports for Mercedes-Benz.

The last five years he specialized in data warehouse architecture, data modeling and business intelligence.

Apart from that he also strives to share his knowledge through conferences, articles, and the Wikipedia pages he maintains.