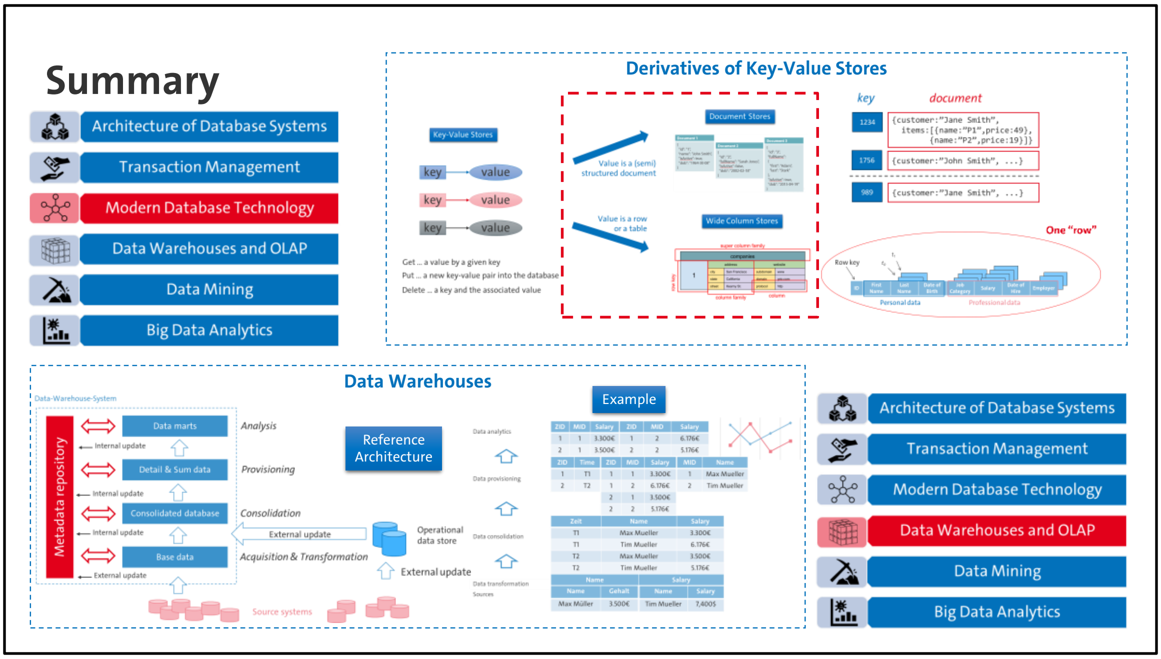
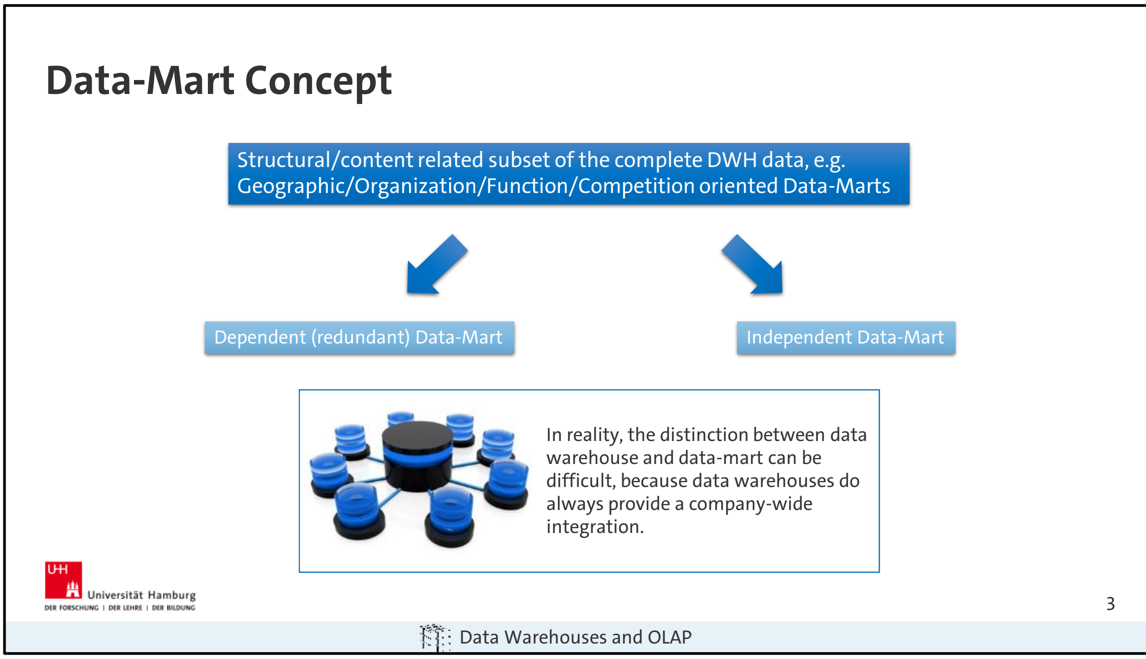

Summary
Architecture of Database Systems
Transaction Management
Modern Database Technology
Data Warehouses and OLAP
Data Mining
Big Data Analytics
Derivatives of Key-Value Stores
Key-Value Stores
key
value
Get ... a value by a given key
Put ... a new key-value pair into the database
Delete ... a key and the associated value
Value is a (semi) structured document
Document Stores
Value is a row or a table
Wide Column Stores
key
document
1234
{customer:"Jane Smith", items:[{name:"P1",price:49}, {name:"P2",price:19}]}
1756
{customer:"John Smith", ...}
989
{customer:"Jane Smith", ...}
One "row"
Row key
Personal data
Professional data
Data Warehouses
Data-Warehouse-System
Metadata repository
Data marts
Internal update
Detail & Sum data
Consolidated database
Base data
External update
Analysis
Provisioning
Consolidation
Acquisition & Transformation
Source systems
Reference Architecture
Operational data store
External update
Data analytics
Data provisioning
Data consolidation
Data transformation
Sources
Example
Architecture of Database Systems
Transaction Management
Modern Database Technology
Data Warehouses and OLAP
Data Mining
Big Data Analytics



Why?

- Read-optimized layer: Data is stored in a denormalized data model for better read performance and better end user usability/understanding
- The Data Mart Layer is providing typically aggregated data or data with less history (e.g. latest years only) in a denormalized data model
- Dividing independent topics, ideally one subject per data mart
- Privacy aspects
- Reduction of data volume → Queries on the whole data warehouse can become a bottleneck
- Performance/Load distribution

DWH 2.0

- Structured and “unstructured” data
- Life cycle of data with different storage areas

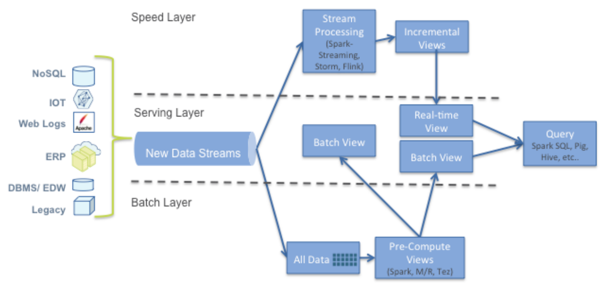
Hot data: High speed, expensive storage for most recent data

Cold data: Low speed, inexpensive, large storage for old data; archival data model with compression

- Metadata s an integral part of the DWH, not just an afterthought

W.H. H. Inmon, Derek Strauss, Genia Neushloss: DW 2.0: The Architecture for the Next Generation of Data Warehousing

Lambda Architecture (Big Data)



Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Batch Layer

- DWH alike, correct & complete
- Output in Read-only DB, complete replacement
- Hadoop/Spark de facto standard

Speed Layer

- Stream processing
- Real-time, latency is king, „batch gap“
- Correct-/completeness minor concern
- Apache Storm, Spark etc.
- Output into fast NoSQL DBs
- Indexes most recently added data

Serving Layer

- Query processing
- Stores outputs, builds views
- Druid, Cassandra, HBase

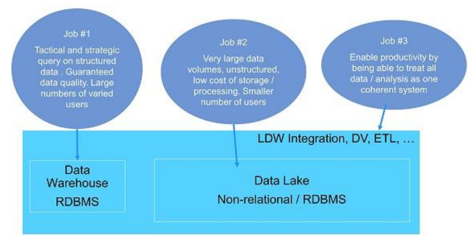
Example: Streaming service analytics

→ Speed layer: we need information about service issues → aggregate only needed to check if any threshold is hit and the system must react

→ Batch layer: analytics about user groups

Logical Data Warehouse

- Focus on Jobs to be done
- **DWH**
  - Query processing on structured data
  - Guaranteed quality
- **Data Lake**
  - Very large volumes of unstructured data
- **Logical Data Warehouse**
  - Provide access to both underlying systems
  - Requires ETL functionality
  - Virtual integration



Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

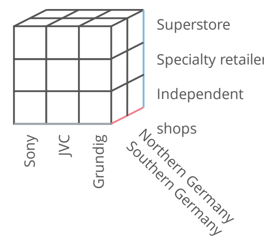
[https://blogs.gartner.com/henry-cook/2018/01/28/the-logical-data-warehouse-and-its-jobs-to-be-done/]

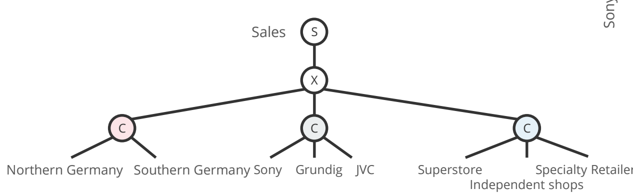
Data Lakes

- With cheap storage costs, people promote the concept of the data lake
  - Combines data from many sources and of any type
  - Allows for conducting future analysis and not miss any opportunity
- Collect everything
  - All data, both raw sources over extended periods of time as well as any processed data
  - Decide during analysis which data is important, e.g., no “schema“ until read
- Dive in anywhere
  - Enable users across multiple business units to refine, explore and enrich data on their terms
- Flexible access
  - Enable multiple data access patterns across a shared infrastructure: batch, interactive, online, search, and others

Multidimensional Modelling

Sales		Sony	JVC	Grundig
Northern Germany	Superstore			
	Specialty retailer			
	Independent shops			
Southern Germany	Superstore			
	Specialty retailer			
	Independent shops			

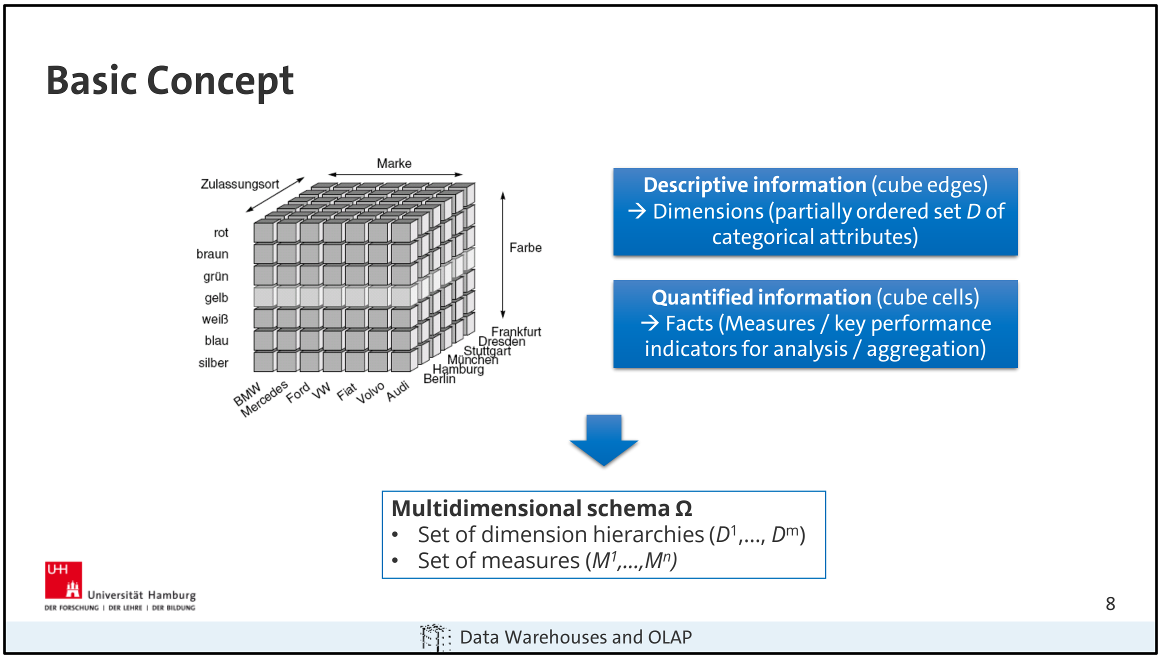




Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Static table also called “summary table”

Motivation: Reporting and interactive Analysis
- OLAP (Online Analytical Processing) on multidimensional data model
- Data Mining: Search for unknown patterns or relations in data
- Visualization



Central data structure: multidimensional cube

- Descriptive data (categorical attributes)
- Quantified data (sum attributes)

Multidimensional modeling

"Predict" analytic patterns of users

- Drill-paths for navigations operators
- Limit to ***meaningful*** aggregation options

Data structures

- **Descriptive information** (cube edges) → dimensions
  - Hierarchies, dimensional attributes
  - Structural basis for selection and aggregation
- **Quantified information** (cube cells) → facts
  - Measures / key performance indicators for analysis / aggregation

Goal

- Orthogonal dimensional descriptions
- Clear separation of measures

Dimensions / Dimension hierarchy

- Partially ordered set D of categorical attributes

$(\{D_1, \ldots, D_n, Top_D\}; \rightarrow)$

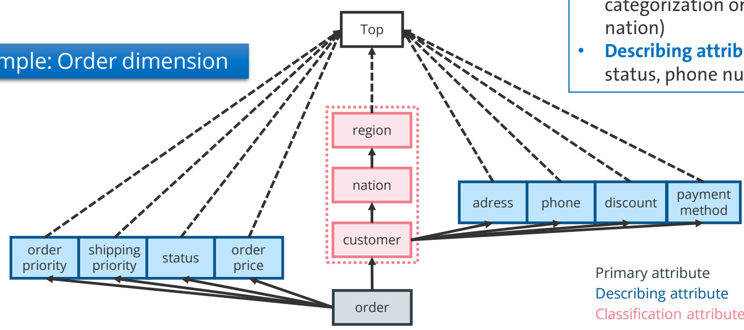
- Top_D is a generic maximum element w.r.t. "→", i.e.
  $\forall i\ (1 \leq i \leq n): D_i \rightarrow Top_D$
- There is a Di with finest granularity, i.e. $D_i \rightarrow D_j$ for all D_j
- "→" denotes the functional dependency
- Partial ordering allows arbitrary parallel hierarchies

!!!Multidimensional Model is conceptual not physical!!

Schema of a Dimension

Roles of categorical attributes
- **Primary attribute**: finest granularity (e.g. order)
- **Classification attribute:** implies multi-stage categorization on value level (e.g. customer, nation)
- **Describing attribute:** additional description (e.g. status, phone number)

Example: Order dimension



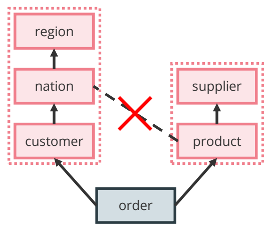
Orthogonality
- No functional dependencies between attributes from different dimensions

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

There is a functional dependency A→B if for all a∈A there is exactly one b∈B, e.g. Germany → Europe, but not Europe → Germany

Schema of a Dimension

Example: Multiple hierarchies

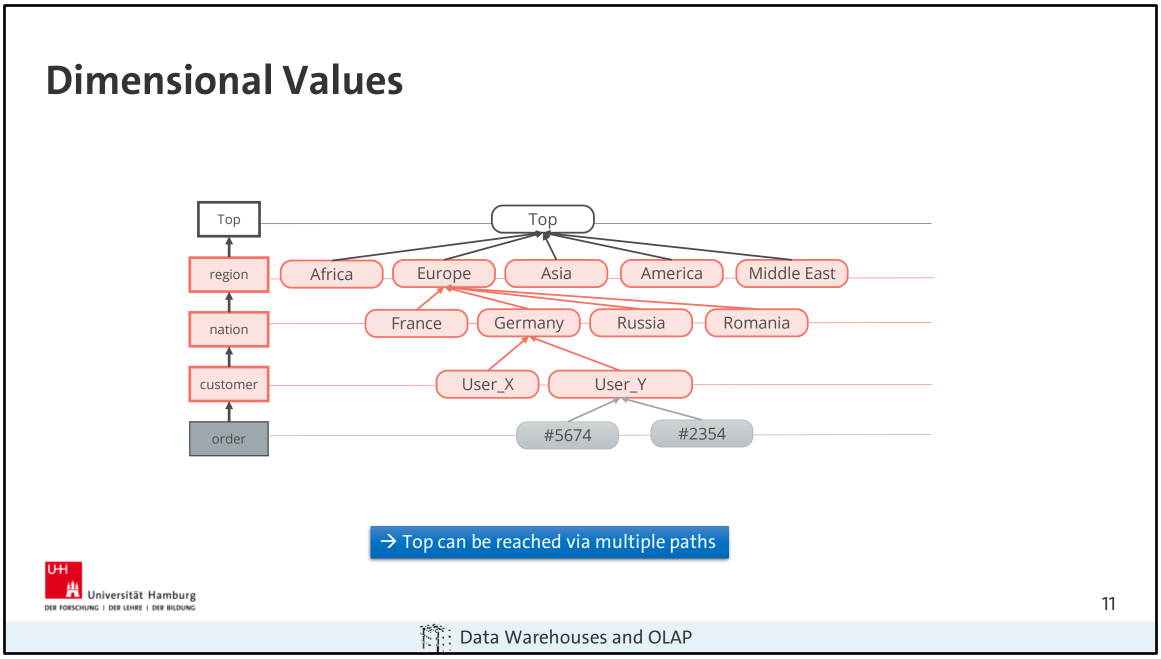


Not modelled here: products sold only in some nations

Primary attribute
Describing attribute
Classification attribute

Orthogonality

- No functional dependencies between attributes from different dimensions



Functional dependencies define tree structure on instances

- Functional dependency corresponds to 1:N-relation!
- Every path from a classification attribute to Top defines a classification hierarchy

Facts and Measures

Quantifying part of a multidimensional data schema

Fact

- Defined as F = (G, SumType)
- Granularity: G = {G1,...,Gn}
- Subset of all categorical attributes of all existing dimensional hierarchies $D_1, \ldots, D_n$
- Summation type: SumTyp $\in$ {FLOW, STOCK, VPU}

Example: Delivered quantity with granularity G={orderNo, partNo, supplierNo}

Measure

- M = (G, f($F_1,\ldots,F_k$), SumType)
- Calculation formula f()
  - Scalar function +,-,•,/,mod etc., e.g. sales tax= quantity • price • tax rate
  - Aggregation functions, e.g. SUM(), MAX(), COUNT(), e.g. SUM(quantity • price)
  - Order-based functions, e.g. cumulation, top-k calculations
- Non-empty subset of all existing Facts $F_1,\ldots,F_k$

Example
Fact: Number and price per product
Scalar measure: Sum of price and sales tax for one product
Aggregated measure: Aggregated price for a completed order
Order-based measure: The k most expensive products of the order

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Summability

Problem

- Not all functions can be summed up (median, quantiles, standard deviation)
- Even simple aggregation functions (SUM, AVG, MIN, MAX, COUNT, ...) cannot be aggregated further at will

Change of granularity

- *$G = (G_1,\ldots,G_n)$ is finer (same) $G' = (G_1',\ldots,G_k')$, i.e. $G \leq G'$ iff, for each $G_j' \in G'$ exists a $G_i \in G$, s.t. $G_i \rightarrow G_j'$*
- Coarsening / refinement of granularity adding / removing categorical attributes
- Example

  (orderNo, partNo) ≤ (customerNo, brand) ≤ (market segment)

Necessary characteristics for summability

- Disjointness, completeness, type compatibility

Characteristics of Summability

Disjointness

→An individual value is considered **exactly once** during calculation

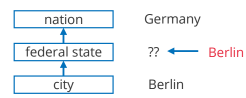
→General: Summability is given if functional dependencies exist between the categories that should be summed up

Number of students	2020	2021	2022	Total
Computer science	15	17	13	**28**
Economics	10	15	11	**21**
Total	**25**	**32**	**24**	**49**

Case 1: sum by year gives correct results
Case 2: sum up number of computer science students
- naive: 15+17+13 = 45 gives wrong result (2 year overlap)
- Assume faculty founding in 2020: 15 + (17 - 15) + (13 - 2) = 28

Completeness

→Measures on higher aggregation-level are completely derived from measures of lower aggregation-levels



Number of restaurants	2010	2011	2012	2013
Dresden	34	38	31	29
Leipzig	123	121	128	131
Chemnitz	21	18	24	27
OTHER	11	13	13	12
Total	**189**	**190**	**196**	**199**

- Number of restaurants in the triangle Dresden-Leipzig-Chemnitz
- OTHER for inns, not located in one of the cities

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

13

Data Warehouses and OLAP

- Possibly, the whole space cannot be captured
  - Example: assume Berlin is not modeled as a federal state (city → federal state)
  - Weak functional dependency (A⇒B): for each a∈A exists at most one b∈B
  - example: city⇒ federal state
- Remove weak functional dependencies
  - NULL, OTHER, dummy values

Summation types

Summability

	FLOW	STOCK: aggregation over temporal dimension?		VPU
		no	yes	
MIN/MAX	✓	✓		✓
SUM	✓	✓	✗	✗
AVG	✓	✓		✓
COUNT	✓	✓		✓

FLOW
Can be aggregated freely, usually measured in <thing> per time

STOCK
States which can be aggregated freely except for a temporal dimension

Value-per-Unit
States which cannot be summed (except for min, max, and avg), e.g. exchange rate
→ Values of different points in time might not be aggregated into one value

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Summation Types

Flow (event at time T)

- Can be aggregated freely
- Examples: order quantity of a certain item per day, number of traffic deaths per month, sales, earnings per year,…

Stock (status at time T)

- Can be aggregated freely, without a temporal dimension
- Items in stock over time → disjointness voided
- Example: number of school kids per month,

Value-Per-Unit (VPU)

- Current states, that cannot be summed
- Use of COUNT()-, MIN()-, MAX()- und AVG() is allowed
- Example: exchange rate, unit costs, …

Exercise Summation Types

	FLOW	STOCK	VPU
Inventory			
Value added tax rate			
Ordered items per day			
#inhabitants per city			
Stock price			
Exams per semester			
SWS (Semesterwochenstunden/ semester hours)			
Exam grade			

The Data Cube

Data cube: C = (DS,M)

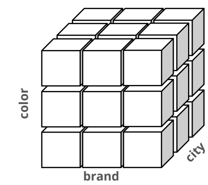
- Set of dimensional hierarchies: $DS = \{D^1, \ldots, D^n\}$
- Set of fact-based measures: $M = \{M^1, \ldots, M^m\}$

Domain of a data cube

- Cartesian product of all value ranges of all attributes of the cube schema

Data cube instance

- All cube cells from the cube domain



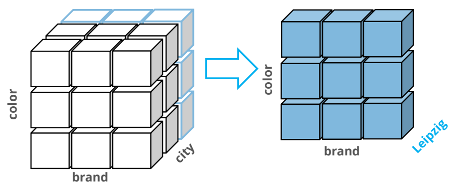
Cube is just a metaphor!

- Almost never, all cells are present (sparsity)
- Non-existent values on the implementation level become NULL or 0 on the model level!

There are usually more than 3 dimensions!

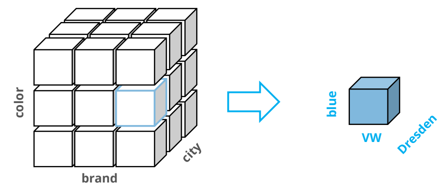
Operations on Data Cubes I

Slicing → Select "slices of a cube"
→ Selection via specification of classification attributes of **one** dimension

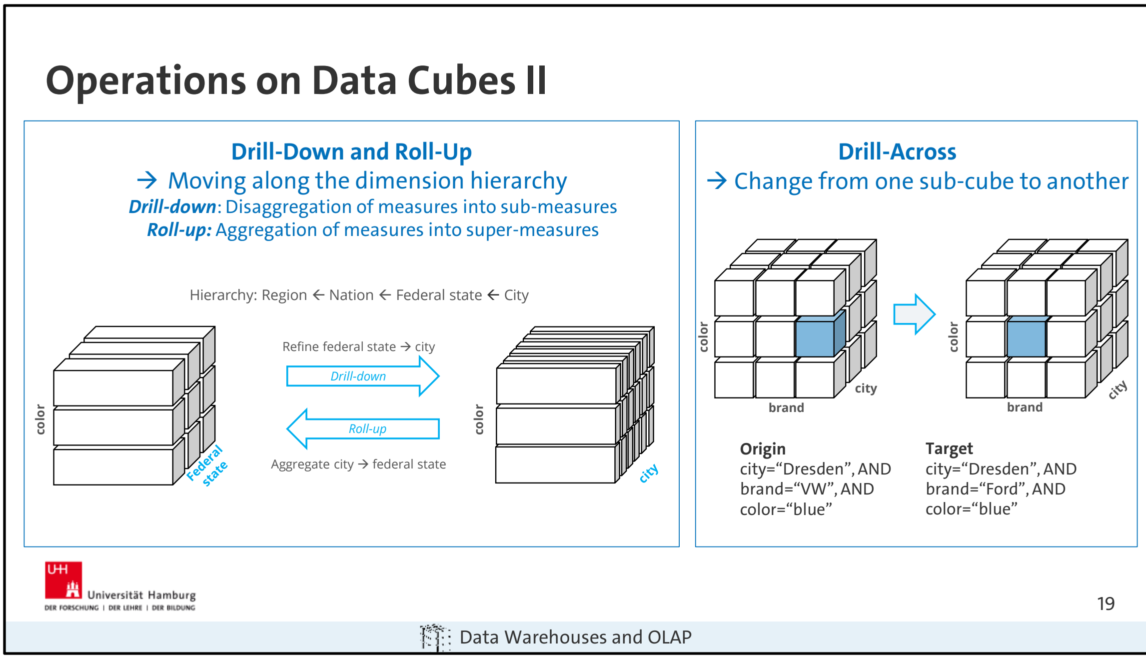


city="Leipzig"

Dicing → Select a sub-cube
→ Selection by specification of classification attributes of **multiple** dimensions



city="Dresden", AND
brand="VW", AND
color="blue"



Drill-Down and Roll-Up

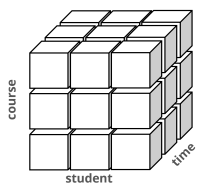
- Changes granularity, not Categorization

Drill-Across

- Dimension stays on the same hierarchy-level, but selection value changes
- Also change of data cube (Join of multiple data cubes)

Exercise Data Cube

- Which operations are described? (slicing, dicing, roll-up, drill-down, drill-across)
- How could you visualize the results?



a) name = “Jane Doh”
b) refine name → first
c) student=“Jane Doh” AND time = “2 pm” AND course = “AD”
d) time = “4 pm” AND course = “Math I”
e) aggregate first → name
f) student=“Jane Doh” AND time = “2 pm” AND course = “AD”
   → student=“Jane Doh” AND time = “10 am” AND course = “AD”

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Data Cube in PostgreSQL

- Install extension: CREATE EXTENSION cube;
- Create a cube: SELECT cube(array[X,Y,Z]); //creates cube with dimensions X, Y, and Z
- Selected functions:
  - Cube_union (cube,cube)
  - Cube_inter(cube,cube) → intersection
  - cube_subset (cube, integer[]) → new cube from existing group using list of dimension indexes, e.g. for dicing
- Rollup is a separate function and a subclause of GROUP BY

rollup (x,y,z)
(assuming the hierarchy is x > y > z) →
(x, y, z)
(x, y)
(x)
()

cube (x,y,z) →
(x, y, z)
(x, y)
(y, z)
(x, z)
(x)
(y)
(z)
()

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

22

https://www.postgresql.org/docs/current/cube.html
https://www.timescale.com/learn/postgresql-extensions-cube

- Cube can also be used in GROUP BY
- Common use of rollup: aggregation of data by year, month, and day

Multidimensional database design

<table>
<tr><th></th><th>Classical relational database design</th><th colspan="2">Multidimensional database design</th></tr>
<tr><td>Conceptual schema (semi-formal)</td><td>Variants of entity-relationship modelling</td><td colspan="2">Different modelling languages, e.g. mE/R, mUML, ADAPT,...</td></tr>
<tr><td rowspan="2">Logical schema (formal)</td><td rowspan="2">Relations with attributes</td><td colspan="2">Data cube: facts and measures</td></tr>
<tr><td colspan="2">Dimensional hierarchy with categorical attributes: classifying and describing attributes</td></tr>
<tr><td>Internal/physical schema</td><td>Memory organisation (primary/secondary indexes, partitioning,...)</td><td>Relational storage (ROLAP): Star/Snowflake schema patterns</td><td>Multidimensional storage (MOLAP): native implementation</td></tr>
</table>

ROLAP = Relational OLAP
MOLAP = Multidimensional OLAP
HOLAP → Hybrid solution