Data Warehousing

Databases, Aarhus University

Ira Assent

Intended learning outcomes

- Be able to
  - Describe the main differences between data warehouses and transactional databases
  - Be able to use characterize main aspects of basic data warehouse design steps

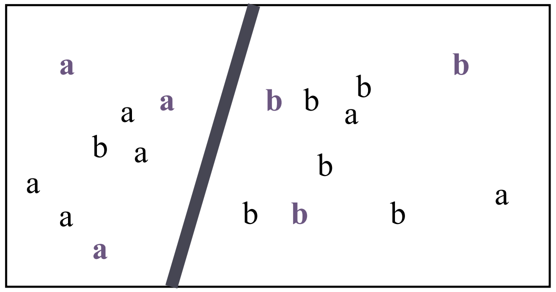
Recap: data mining

1. {butter, bread, milk, sugar}
2. {butter, flour, milk, sugar}
3. {butter, eggs, milk, salt}
4. {eggs}
5. {butter, flour, milk, salt, sugar}

- Discovery of new information (patterns) from data
- Association rule
  - Sugar ⇒ Butter: support 3/5=60%, confidence 1=100%
  - Find frequent itemsets (item combinations exceeding minimum support)
    - based on apriori principle: any subset of a frequent itemset must be frequent
    - 1-itemsets, then 2-itemsets, 3-itemsets, and so on, using only frequent shorter ones
  - Create rules A ⇒ (X – A) for each frequent itemset X and each subset A of X
    - Keep those that exceed minimum confidence
- Clustering: group objects into sub-groups (clusters)
  - "maximize" intra-class similarity and "minimize" interclass similarity
  - K-means: given *k*, form *k* groups so that sum of distances between mean elements minimal
    - From random initialization, keep recomputing mean of clusters, and assign points to closest mean until converged
- Classification: Learn model to describe classes and make predictions
  - Decision Tree
    - Flow-chart like graphical representation
    - Nodes are attributes, branches are outcomes, leaves are class predictions

Classification

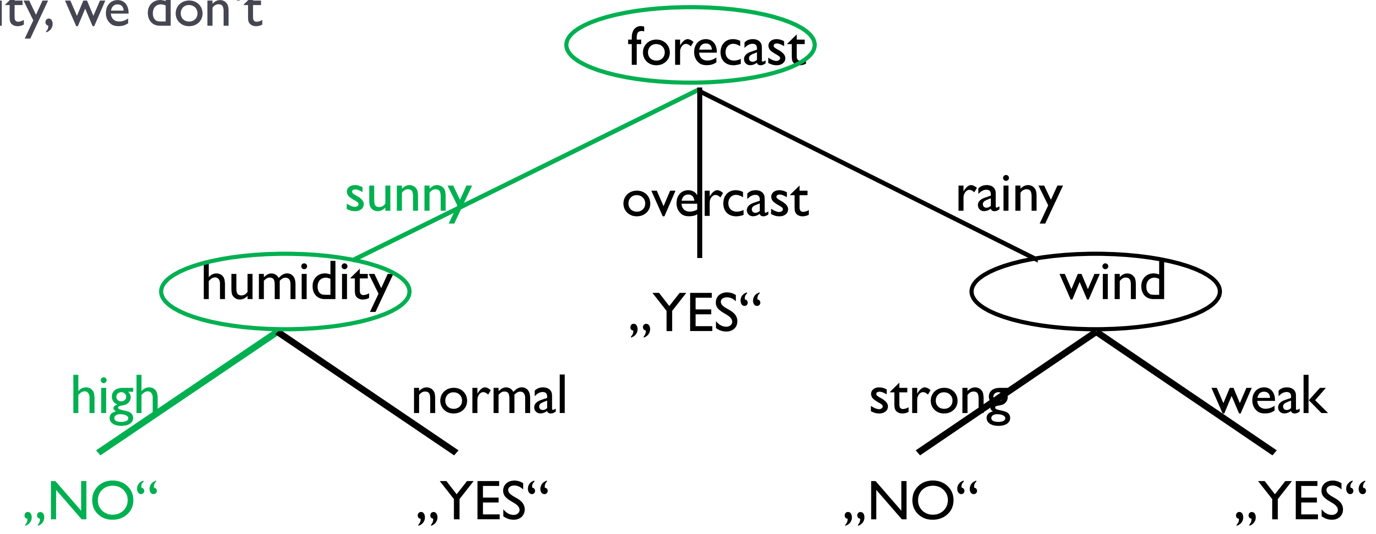
- Learning a model able to describe different classes of data
- Supervised as the classes to be learned are predetermined
- Class labels are known for a small set of "training data": Find models/functions/rules (based on attribute values of the training examples) that
  - describe and distinguish classes
  - predict class membership for "new" objects



- Applications
  - Classify gene expression values for tissue samples to predict disease type and suggest best possible treatment
  - Automatic assignment of categories to large sets of newly observed celestial objects
  - Predict unknown or missing values (→ KDD pre-processing step)
  - . . .

Decision Tree Classifiers

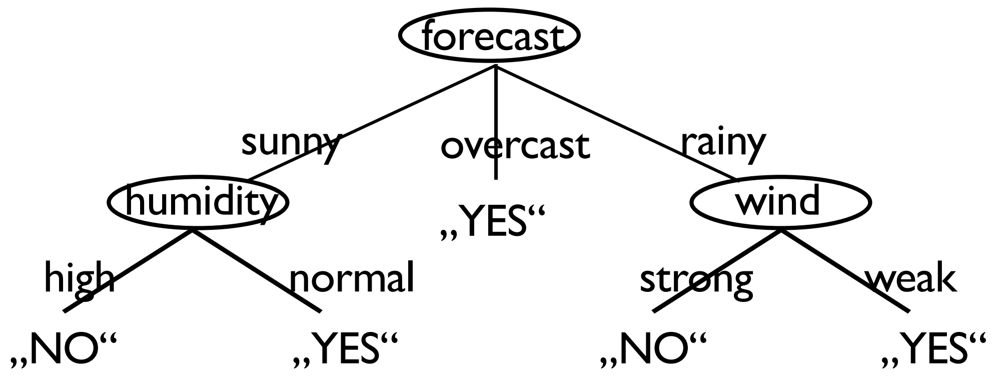
- Decision Tree
  - Flow-chart like graphical representation
  - Intuitive and interpretable classification model
- Structure
  - Nodes are attributes, branches are outcomes, leaves are class predictions
- Example: are we going to play tennis?
  - Let‘s check the weather: if today‘s forecast is sunny, and we have high humidity, we don‘t



Learning Decision Tree Classifiers

- Idea: make use of **training data**
  - „historical“ data for which we know the outcome (play tennis NO/YES)
  - Many records, each of which has attribute values for a number of attributes and a value for the class (target attribute: tennis)
  - To build the tree:
    - Start top-down from the root
      - Group data according to values for different attributes
        - E.g. forecast sunny/overcast/rainy gives two groups, wind strong/weak gives two groups
- Training data set:

day	forecast	temperature	humidity	wind	tennis
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rainy	mild	high	weak	yes
5	rainy	cool	normal	weak	yes
6	rainy	cool	normal	strong	no
7	. . .	. . .	. . .	. . .	. . .



Building Decision Trees

- Tree is created top-down
  - We greedily try to reduce our uncertainty about the class outcome (YES/NO)
    - Root is decision on the attribute that helps most in this regard

  - Training examples T recursively partitioned into $T_1, T_2, \ldots, T_m$
    - Entropy for k classes with frequencies p_i (information theory: measure of uncertainty)

$$entropy(T) = -\sum_{i=1}^{k} p_i \cdot log_2 p_i$$

    - Compute the information gain of a split using an attribute, such as humidity, by comparing the entropy before the split with the entropy of the split
    - Always pick the split that reduces uncertainty most
      - highest information gain

$$information\ gain(T, A) = entropy(T) - \sum_{i=1}^{m} \frac{|T_i|}{|T|} \cdot entropy(T_i)$$

Example: building Decision Trees

$$entropy(T) = -\sum_{i=1}^{k} p_i \cdot log_2 p_i$$

$$information\ gain(T, A) = entropy(T) - \sum_{i=1}^{m} \frac{|T_i|}{|T|} \cdot entropy(T_i)$$

Day	Outlook	Temp	Humid	Wind	Play?
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

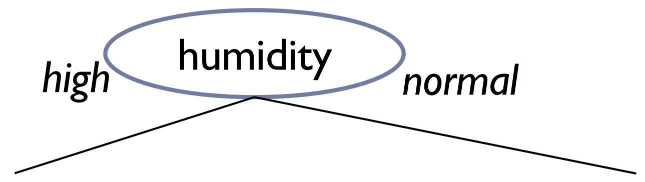
Example: building Decision Trees

$$entropy(T) = -\sum_{i=1}^{k} p_i \cdot log_2 p_i$$

$$information\ gain(T, A) = entropy(T) - \sum_{i=1}^{m} \frac{|T_i|}{|T|} \cdot entropy(T_i)$$

- Start from the root: all training data
  - What is the uncertainty?
  - Entropy (T) uses number of samples in each class, 9 yes, 5 no
    - Entropy(T)=-((5/14)*log(5/14)+(9/14)*log(9/14))=0.940
  - Which possible splits can we do?
    - For each attribute compute entropy based on splits it generates
    - Compare resulting information gain for all four attributes
      - Pick the one with highest information gain
    - For example: humidity, wind
  - Assume for sake of example only, we pick humidity, then below humidity, we check all three remaining attributes to determine the best split – both for the "high" branch and for the "normal" branch outcome likely to differ

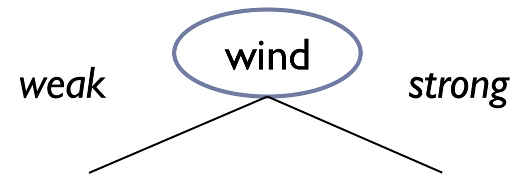
Day	Outlook	Temp	Humid	Wind	Play?
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No



3 „YES“ 4 „NO“, prob. 3/7 vs. 4/7
Entropy = 0.985

6 „YES“ 1 „NO“, probl. 6/7 vs. 1/7
Entropy = 0.592

$$IG(T, hum) =$$
$$0.94 - \frac{7}{14} * 0.985 - \frac{7}{14} * 0.592 = \boxed{0.151}$$



6 „YES“ 2 „NO“
Entropy = 0.811

3 „YES“ 3 „NO“
Entropy = 1.0

$$IG(T, wind) =$$
$$0.94 - \frac{8}{14} * 0.811 - \frac{6}{14} * 1.0 = 0.048$$

Evaluation of Classifiers

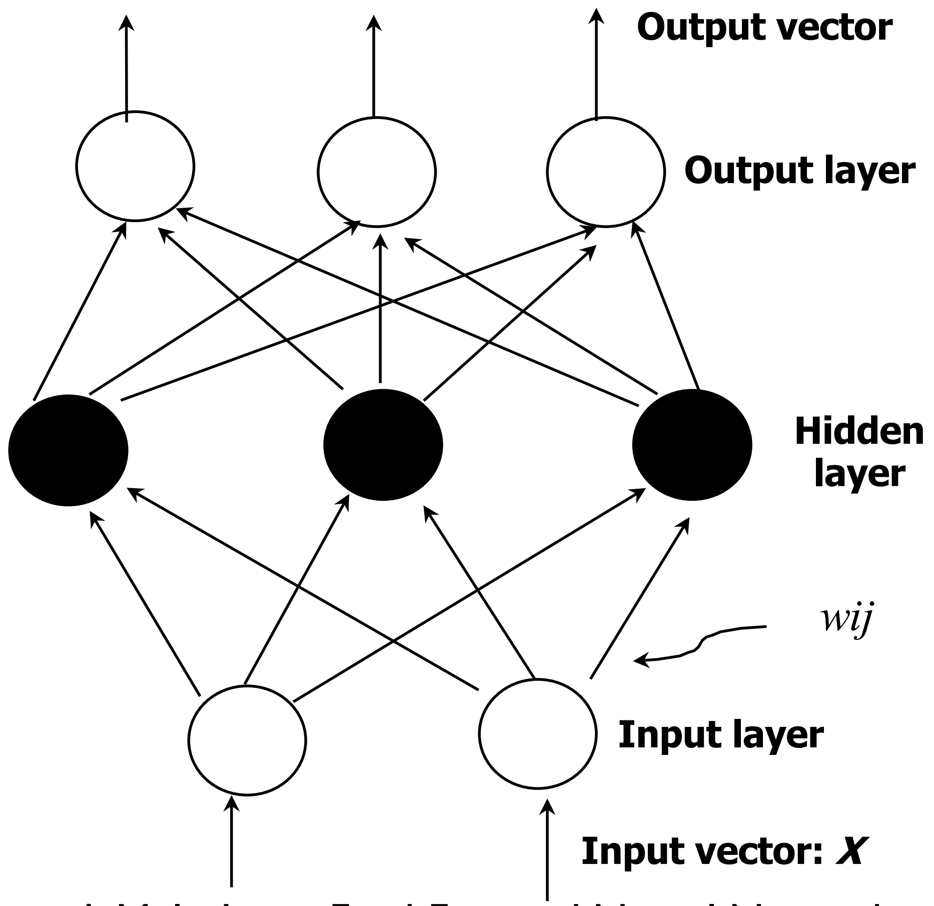
- Classification Accuracy
  - Predict class label for each object o
  - Determine the fraction of correctly predicted class labels:

$$classification\,accuracy = \frac{count(\text{correctly predicted class label})}{count(o)}$$

  - Classification error = 1 – classification accuracy
- Overfitting: accuracy worse on entire data than on training data
    - bad quality of training data (noise, missing values, wrong values)
    - different statistical characteristics of training data and test data
- Train-and-Test: decomposition of data set into two partitions
  - Training data to train the classifier
    - Model construction using information also class labels
  - Test data to evaluate the classifier
    - temporarily hide class labels, predict them and compare

Neural Networks

- Neural network: set of interconnected nodes inspired by human brain (not model of brain!)
  - Nodes in one layer connected to nodes in next layer
  - Node connections have weights
- Used for supervised learning and unsupervised clustering
- Recently dramatic improvements in performance
  - Big Data: lots of training data
  - New training methods and network architectures
- The output of a neural network is quantitative and not easily understood

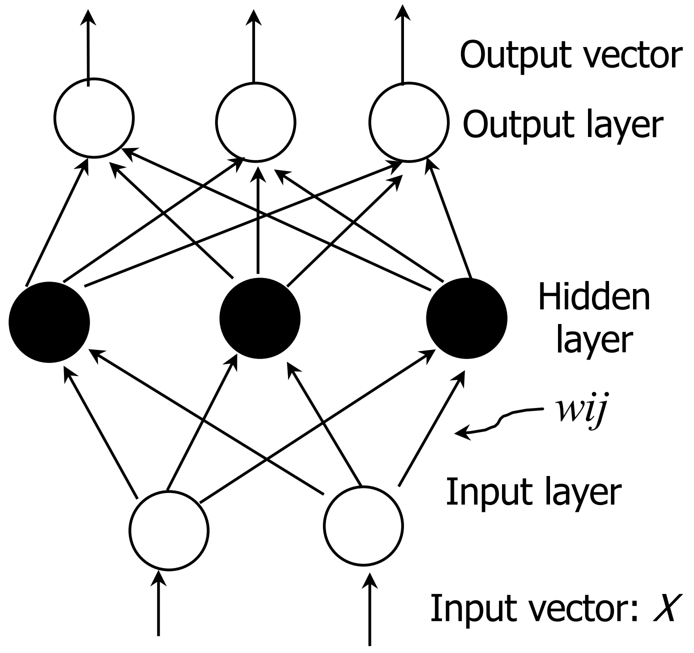


A Multi-Layer Feed-Forward Neural Network

How A Multi-Layer Neural Network Works

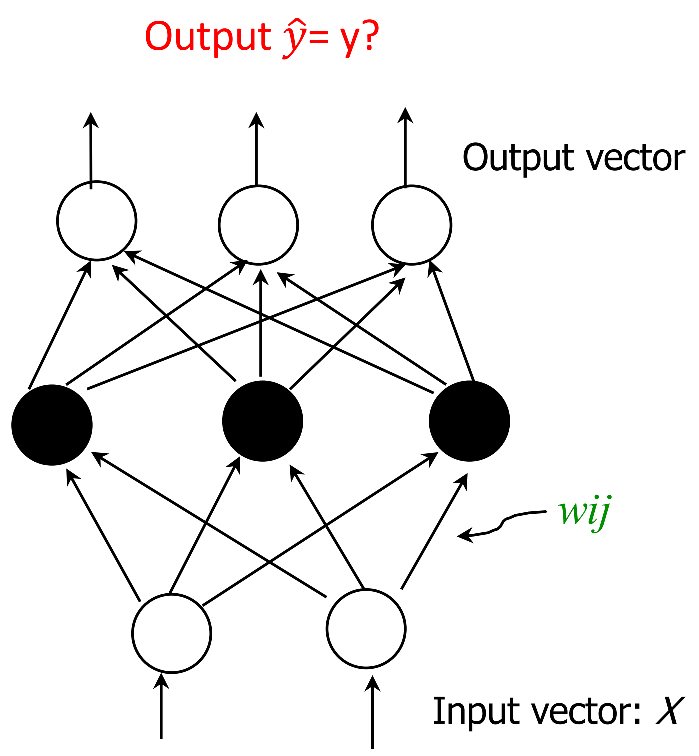
- The **inputs** to the network correspond to the attributes measured for each training tuple
  - Inputs are fed simultaneously into the units making up the **input layer**
  - They are then weighted and fed simultaneously to first **hidden layer, then to second etc**
- The number of hidden layers is arbitrary, many layers: deep learning
- The weighted outputs of the last hidden layer are input to units making up the **output layer**, which emits the network's prediction
- The network is **feed-forward**: None of the weights cycles back to an input unit or to an output unit of a previous layer
- From a statistical point of view, networks perform **nonlinear regression**: Given enough hidden units and enough training samples, they can closely approximate any function

$$w_j^{(k+1)} = w_j^{(k)} + \lambda(y_i - \hat{y}_i^{(k)})x_{ij}$$



Training neural networks

- Iteratively process a set of training tuples and compare the network's prediction with the actual known target value
- For each training tuple, the weights are modified in the opposite direction of the error (gradient descent)
- minimize a **loss function** (error) between the network's prediction and the actual target
  - Use backpropagation to compute gradient on loss function



Why are NN popular now?

- Deep learning
  - Artificial neural networks with many layers
  - Can learn very complex functions
  - Impressive performance on some tasks
    - E.g. object recognition: better than “average human”
- Big Data
  - We have enough data to train large / complex networks
- Modern hardware
  - We have the computational power to train large networks with lots of data
- New architectures and training algorithms
  - We have new modules (ways of connecting nodes) for specific learning goals
  - We have efficient and effective training algorithms

So, what's ChatGPT then – part 2?

- Machine learning based NLP (natural language processing)
- Chatbot by OpenAI, based on GPT language model https://chat.openai.com/
  - Language model: learn to predict which text(piece) should be next
- Based on some text input, learns advance statistics
  - Key ingredients: lots of text (Big Data), lots of hardware (training the model), and powerful machine learning models (**Deep Learning**, Reinforcement Learning)
    - Reinforcement learning: in particular studied in robotics / agent systems:
      - explore and exploit an environment: take an action, receive a reward based on how good your new state is

Further Data Mining / Machine Learning Methods

- Sequential pattern analysis
  - find **subsequences** from given set of sequences that exceed minimum support
    - Sequence S_1, S_2, S_3, .. customer purchasing itemset S_1 likely to buy S_2, then S_3,…
    - Temporal order relevant: buy baby milk, then buy children's food; not so much the other way around
- Time Series Analysis
  - Identify the price trends of a stock or mutual fund
  - Generally temporal trends of values
- Regression
  - Regression equation estimates dependent numeric variable Y using set of independent numeric variables x_i and set of constants: $Y=f(x_1,x_2,\ldots,x_n)$
    - Linear regression: f linear in domain variables x_i,

- Machine Learning course (Bachelor, 3rd year)
- Cluster Analysis (Master)
- Data Mining course (Master)
- Advanced Data Management and Analysis course (Master)
- Computational Learning Theory (Master)
- Natural Language Processing (Master)

What is supervised learning?

A. Clustering based on total distance.
B. Association rule mining with fixed minimum support.
C. Model creation based on class label information.
D. Learning with user input.

Purpose of Data Warehousing

- Tools that provide decision makers with information to make decisions quickly and reliably based on historical data
- Traditional databases optimized for querying and updating, typical workload of transactions that access limited parts of the data
    - Operational / transactional (**OLTP – online transactional processing**)
        - `SELECT * FROM Customers WHERE Name='Chris';`
        - `UPDATE Products SET price=price/2 WHERE 42<Pid<117;`
- Most of the time the data warehouse users need only read access
    - need the access to be fast over a large volume of data
- Analytics
    - Which types of Customers do I have?
    - Who buys products when they are on sale?
    - Data Mining (clustering, extracting association rules, learning a classifier)
- Most of the data required for **data warehouse** analysis
    - multiple databases
    - analysis are recurrent and predictable
    - design specific software to meet the requirements

Introduction, Definitions, and Terminology

W. H. Inmon characterized a data warehouse as:

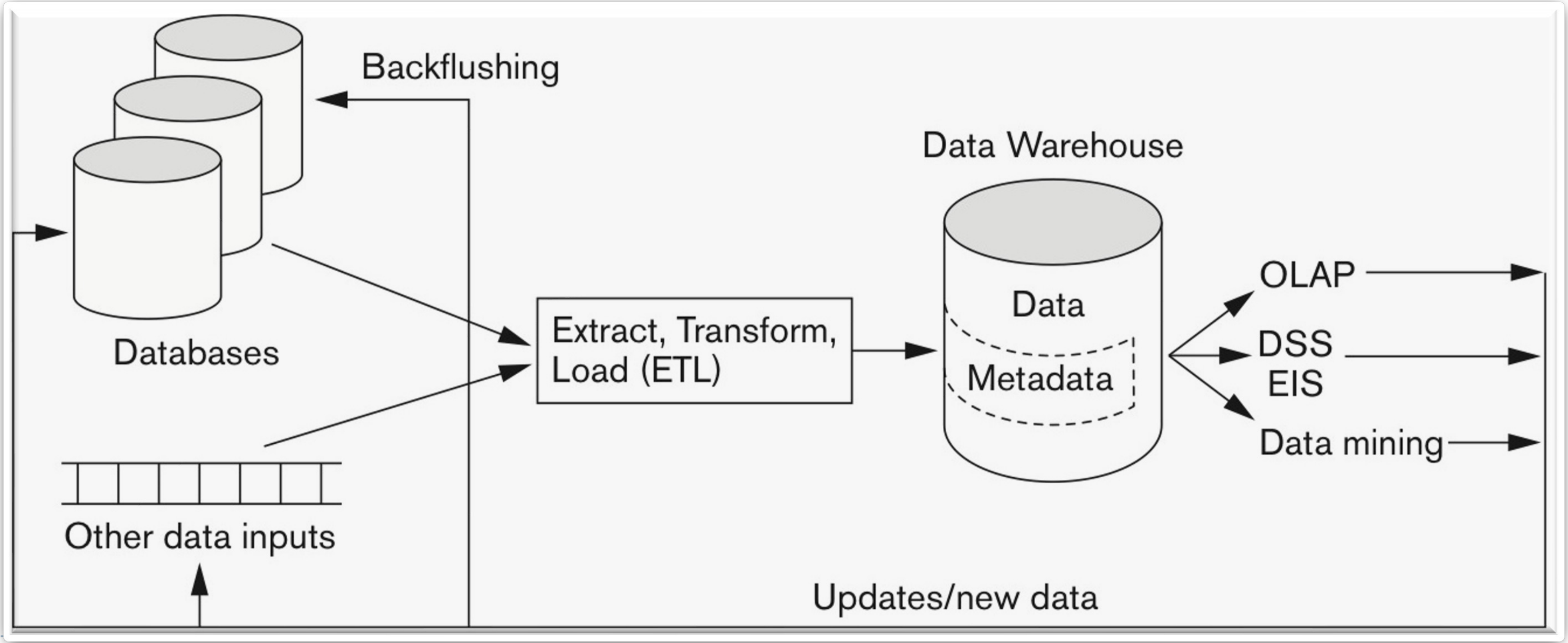
- **"A subject-oriented, integrated, nonvolatile, time-variant collection of data in support of management's decisions."**

Data warehouse supports

- **OLAP** (Online Analytical Processing)
  - analysis of complex data from the data warehouse
- **DSS** (Decision Support Systems) also known as EIS (Executive Information Systems) or MIS (Management Information systems)
  - supports organization's leading decision makers
- **Data Mining**
- Ad hoc queries
  - Interest driven, posed by user once or seldomly
- Canned queries
  - A-priori defined queries with parameters, recur frequently
- Typically business-oriented
  - Obtain overview, extract reports, study parts of the data in detail

Conceptual Structure of Data Warehouse

- Data Warehouse processing involves
  - Cleaning and reformatting of data
  - OLAP
  - Data Mining

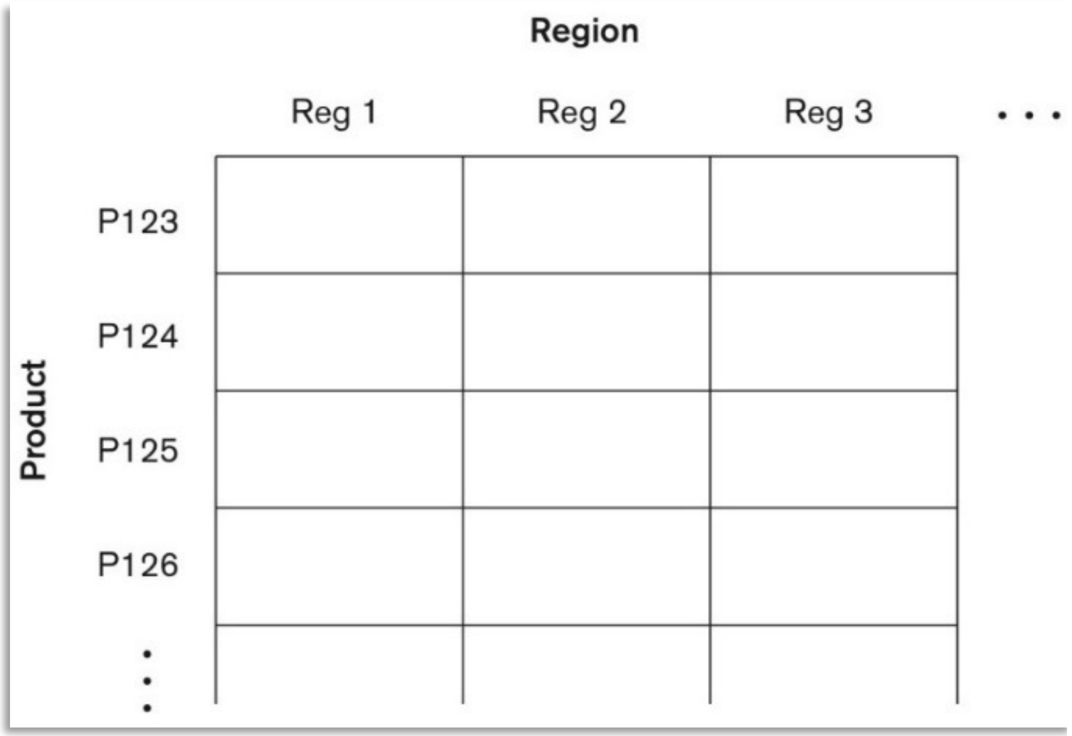


In OLAP, we expect

1. Almost exclusively write access; low selectivity
2. Almost exclusively read access; low selectivity
3. Almost exclusively write access; high selectivity
4. Almost exclusively read access; high selectivity

Data Modeling for Data Warehouses

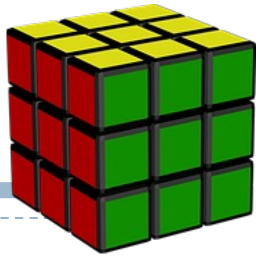
- **Measures** on what we would like information about
  - E.g. sales
- Relate these measures to different **attributes**
  - E.g. regions, products
- Can be represented as two-dimensional spreadsheet or table
  - E.g. columns represent regions, rows products; values sales per region/product



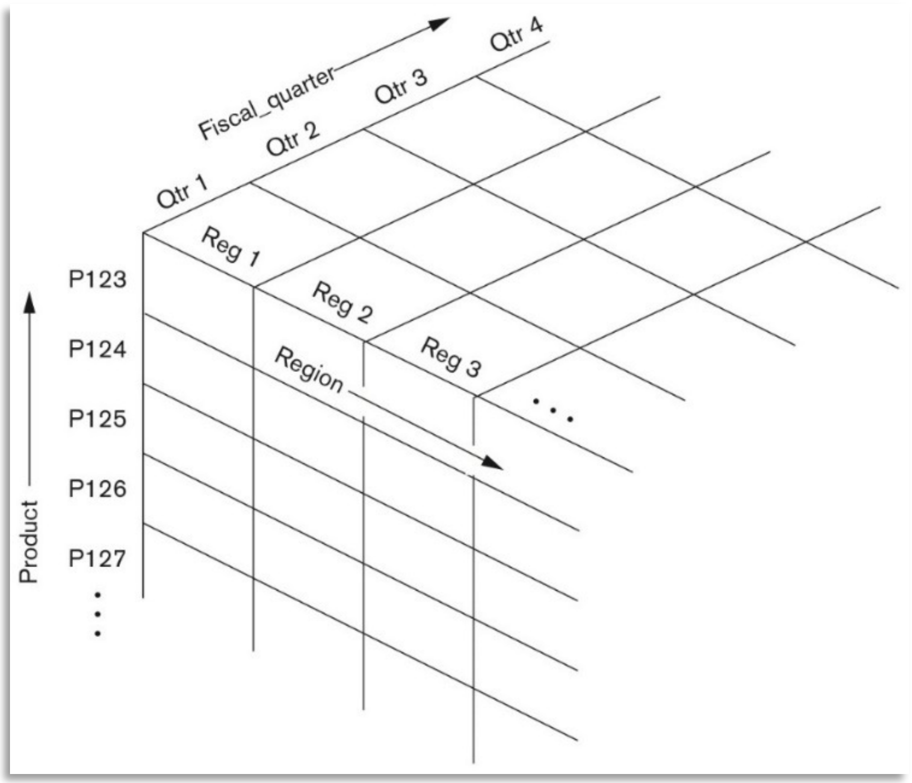
Region

Product	Reg 1	Reg 2	Reg 3	. . .
P123				
P124				
P125				
P126				
⋮				

Multidimensional models

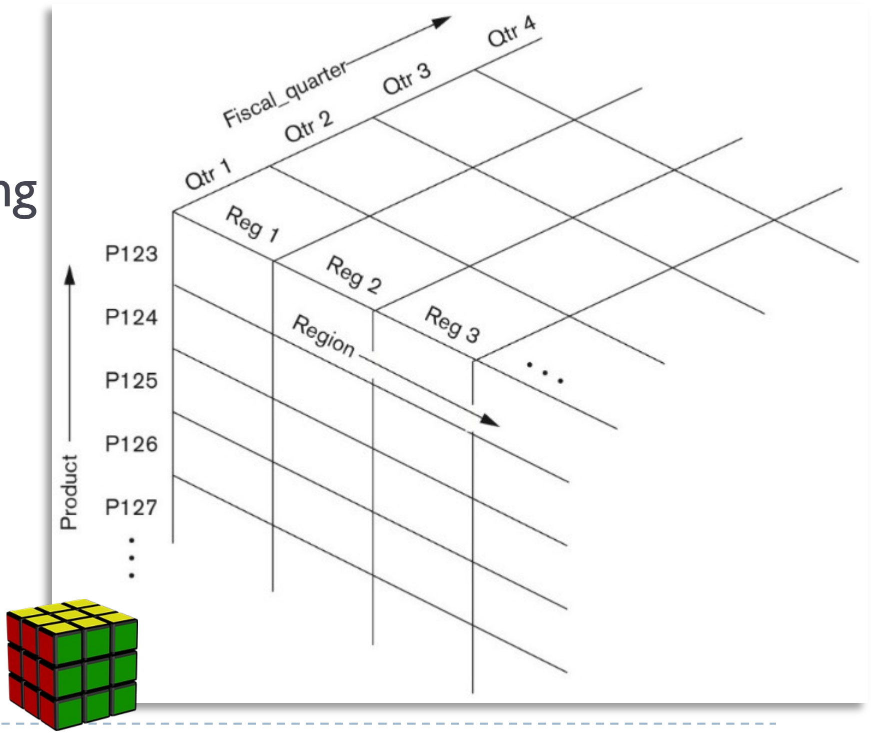


- Querying performance for analysis in a multi-dimensional data storage model is much more efficient
- Data warehouses can take advantage of this feature as generally these are
  - Non volatile
  - The degree of predictability of the analysis that will be performed on them is high
- Example:
  sales per region/product/time period



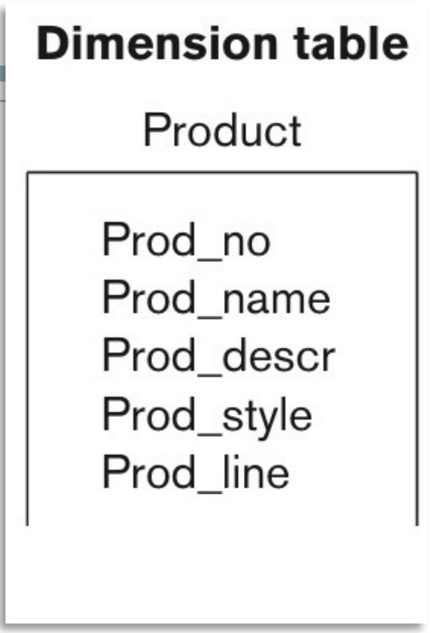
Data Modeling for Data Warehouses

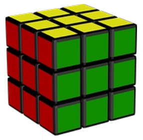
- Advantages of a multi-dimensional model
  - Multi-dimensional models lend themselves readily to hierarchical views in what is known as roll-up display and drill-down display
    - Moving up and down a hierarchy on a dimension
    - E.g. sales per region in a fiscal year vs. sales per region in a fiscal quarter
  - The data can be directly queried in any combination of dimensions, bypassing complex database queries

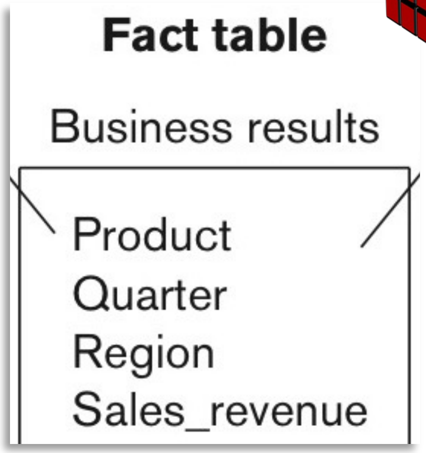


Multi-dimensional Schemas

- Multi-dimensional schemas are specified using
  - **Dimension table**
    - It consists of tuples of attributes of the dimension
    - E.g. product 1, product 2, …, product n
  - **Fact table**
    - Each tuple is a recorded fact
    - This fact contains some measured or observed variable(s)
    - identifies it with pointers to dimension tables
    - fact table contains the data, and
    - the dimensions to identify each tuple in the data

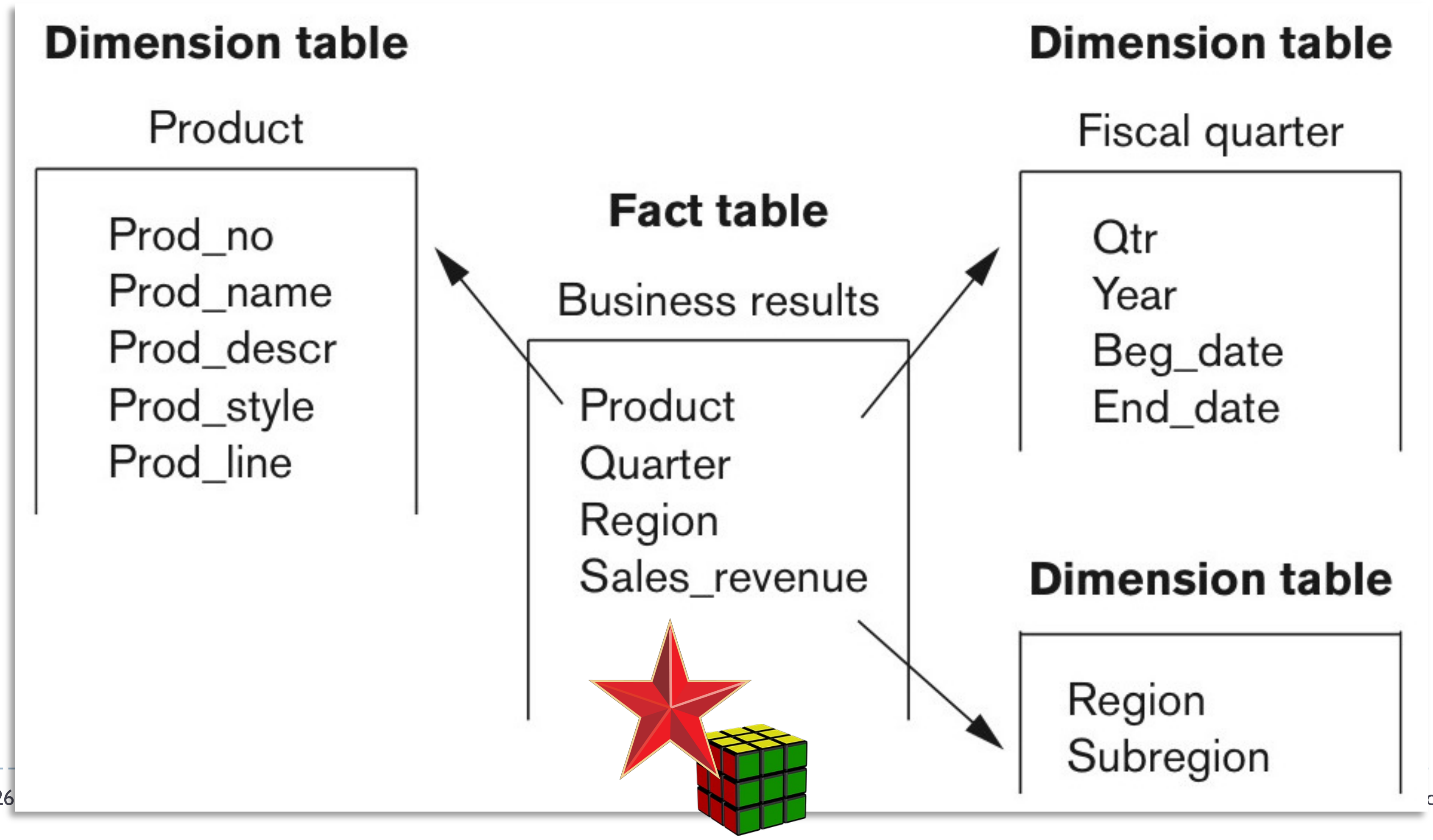






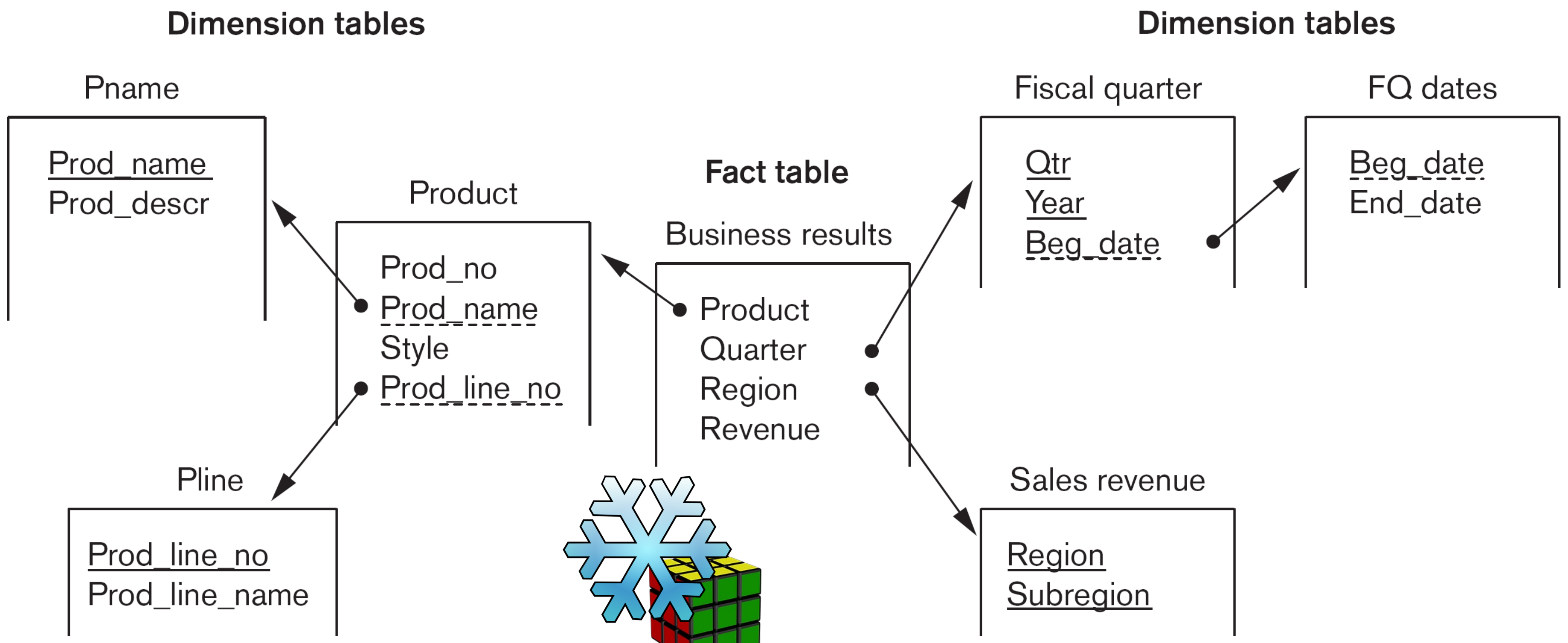
Multi-dimensional Schemas: Star Schema

- **Star schema**
  - Consists of a fact table with a single table for each dimension



Multi-dimensional Schemas: Snowflake schema

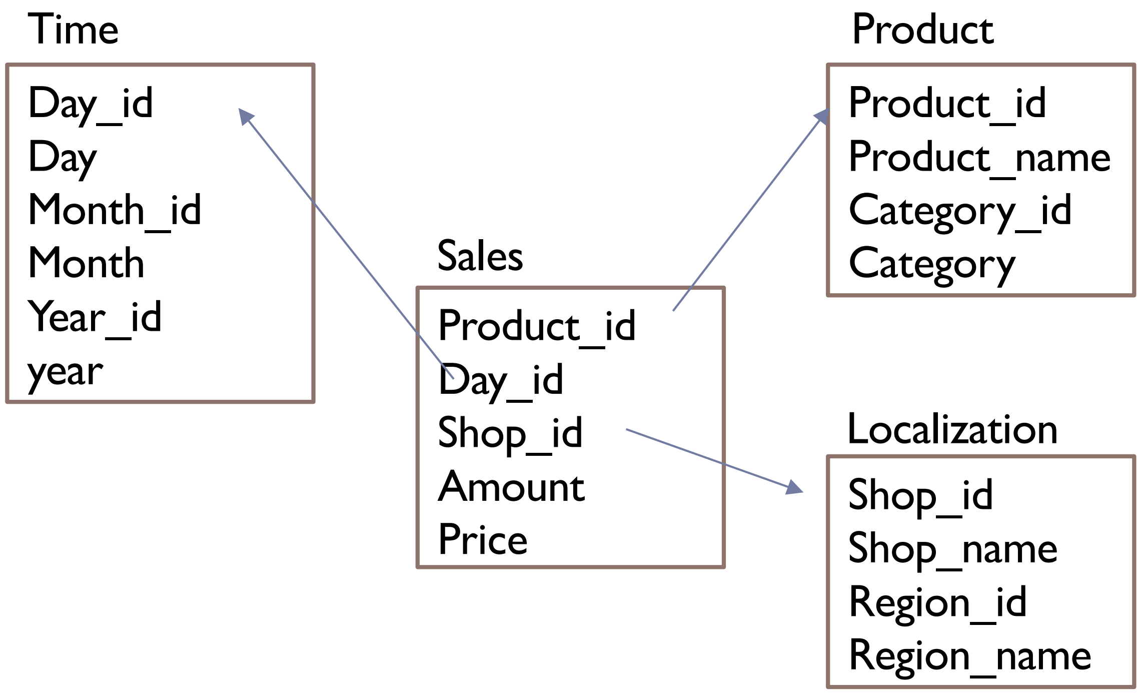
- **Snowflake Schema**
  - Variation of star schema
  - dimensional tables from a star schema are organized into a hierarchy by normalizing them



Facts in SQL

How do we write an SQL query to find the data for the cells in a multi-dimensional table?

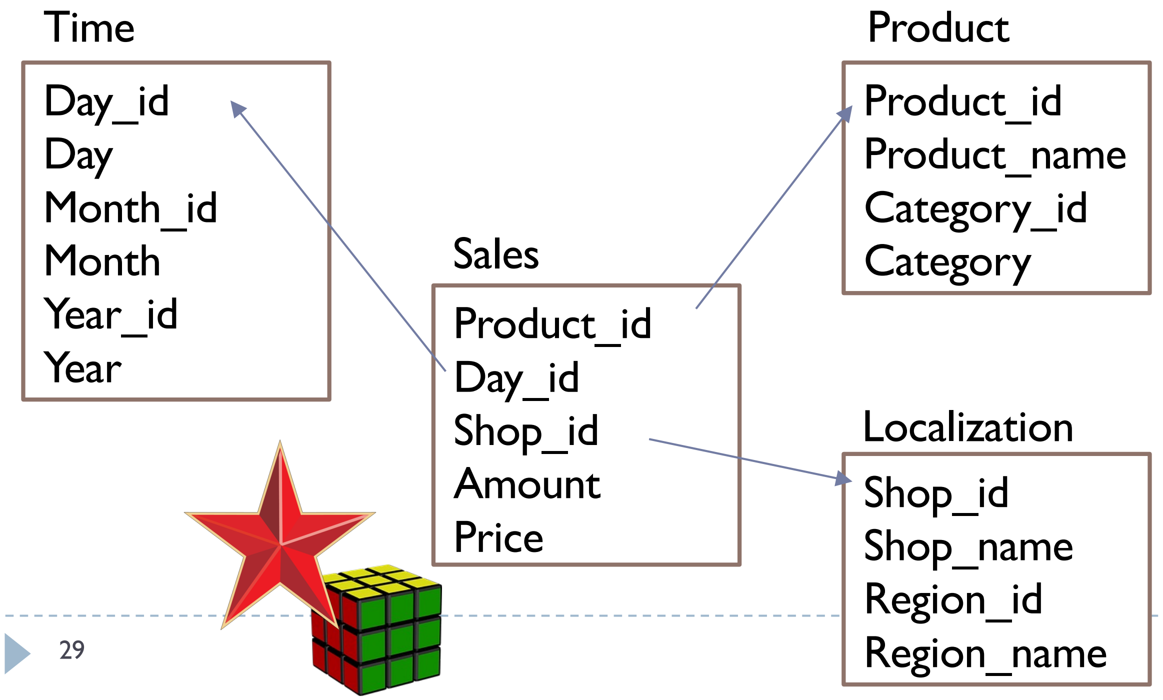
1. Using arithmetics and WHERE conditions
2. Using aggregates and GROUP BY
3. Using joins and projections
4. Using the Cartesian product and ORDER BY



	shop 1	shop 2	…	shop n
year 1	sales shop1 year1	sales shop 2 year1		sales shop n year 1
year 2	sales shop1 year 2	sales shop 2 year 2		sales shop n year 2
…	…	…		…
year m	sales shop1 year m	sales shop 2 year m		sales shop n year m

Star schema in Relational DBMS

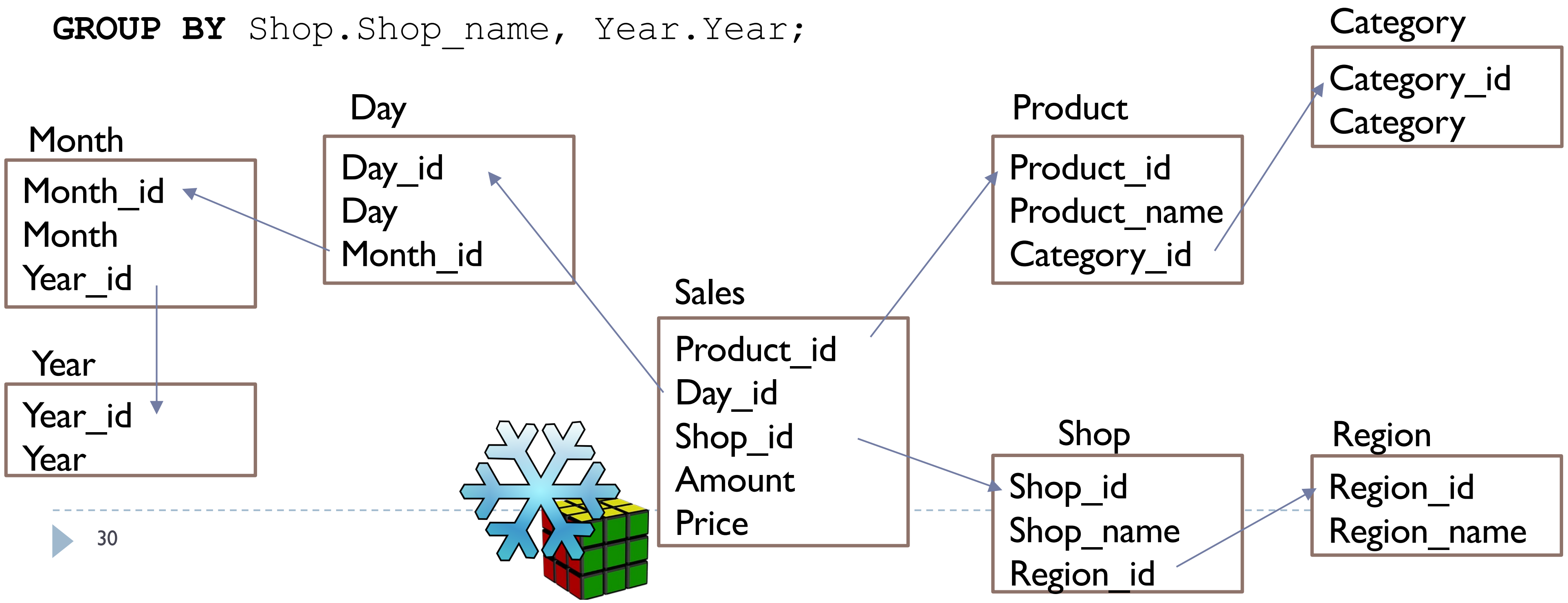
```
SELECT Localization.Shop_name, Time.Year, SUM(Amount*Price)
FROM Sales, Product, Time, Localization
WHERE  Product.Category='Dairy' AND
       Product.Product_id=Sales.Product_id AND
       Time.Day_id=Sales.Day_id AND
       Localization.Shop_id=Sales.Shop_id
GROUP BY Localization.Shop_name, Time.Year;
```



Snowflake schema in RDBMS

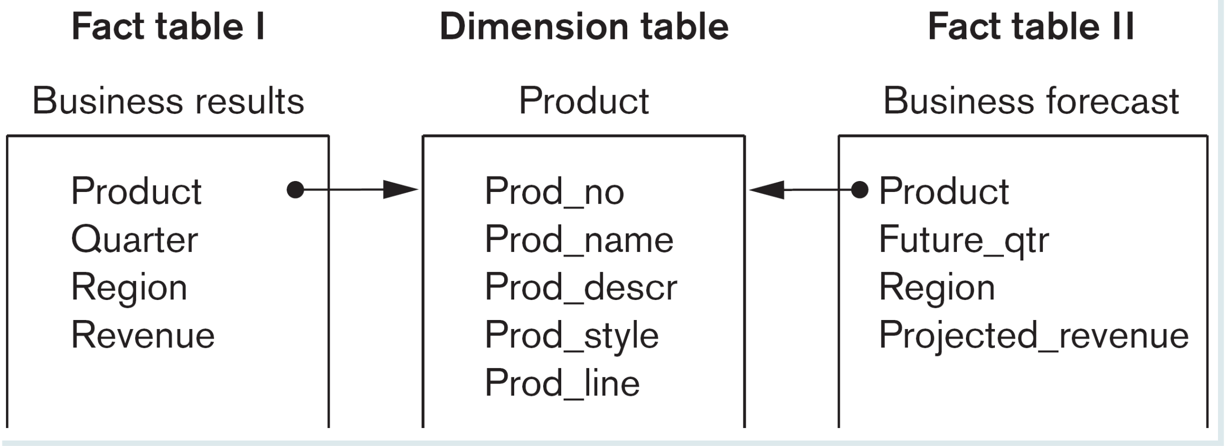
- Snowflake schema incurs complex joins

```
SELECT Shop.Shop_name, Year.Year, SUM(amount*price)
FROM Sales, Product, Category, Day, Month, Year, Shop, Region
WHERE  Category.Category='Dairy' AND
       Category.Category_id=Product.Category_id AND
       Product.Product_id=Sales.Product_id AND
       Day.Day_id=Sales.Day_id AND Day.Month_id=Month.Month_id AND
       Month.Year_id=Year.Year_id AND Shop.Shop_id=Sales.Shop_id
GROUP BY Shop.Shop_name, Year.Year;
```

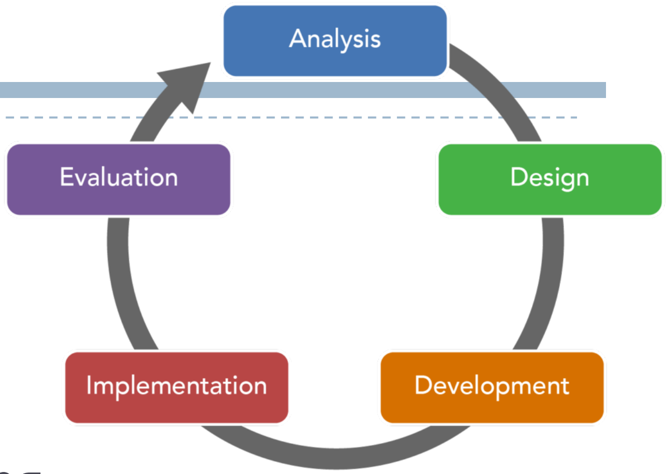


Fact Constellation

- Set of tables that share some dimension tables
  - Can be viewed as collection of many star schemas
  - Also called galaxy schema
- More complex than star or snowflake because of multiple fact tables (multiple aggregations)
  - Flexible, but hard to manage and support



Building a Data Warehouse

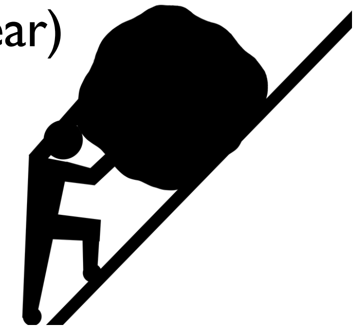


- Good design, as in traditional databases, requires an analysis of the anticipated use of the warehouse
  - i.e., what are typical analyses – the queries
  - Still, the design should support ad-hoc querying
    - E.g. accessing data with any combination of values for meaningful attributes in dimension/fact tables
  - An appropriate schema should be chosen that reflects the anticipated usage
- Design of data warehouse involves
  - Acquisition of data for the warehouse
  - Ensuring that data storage meets query requirements efficiently
  - Giving full consideration environment of data warehouse

ETL – data acquisition

→ A large part of the effort in Data Warehousing lies in **ETL**

- **Extract, transform, load**
- Often greatly underestimated time and effort
- Process of inserting data from the transactional database(s)
  - Different source databases with different schemas
    - Different semantics (e.g. different "years": fiscal vs. calendar year)
- Cleaning
  - Validity and quality of the data
    - Erroneous and incomplete data: difficult to automate
      - E.g. domain constraints
  - Corrected data can be backflushed to the transactional database (e.g. incorrect customer address)
- Converted to data model of Data Warehouse
- Loading of large data volumes is challenging in itself
  - Typically incrementally: go offline for a particular time at regular intervals



Views and DWs?

AUVOTE

1. Yes, views can be used to implement a DW
2. Yes, updatable materialized views can be used to implement a DW
3. No, views are just extracts from the database
4. No, views do not provide the same functionality

Warehouse vs. Data Views

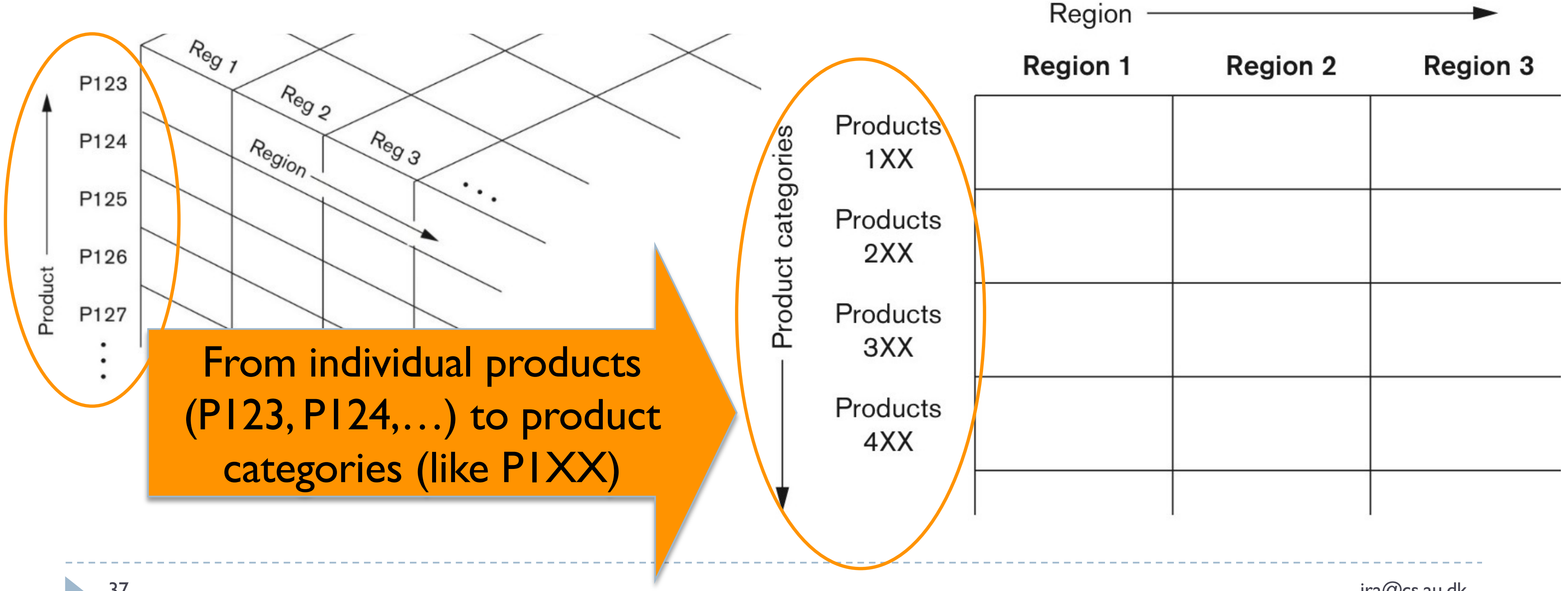
- Views and data warehouses are alike in that they both have read-only extracts from the databases
- However, data warehouses are different from views in that
  - Data warehouses exist as persistent storage instead of being materialized on demand
  - Data warehouses are usually not relational, but multi-dimensional
  - Data warehouses can be indexed for optimization
  - Data warehouses provide specific functionality support
  - Data warehouses deal with huge data volumes generally contained in more than one database

Navigating a Data Warehouse

- Functionality to navigate and study data
  - **Roll-up**: Data is summarized with increasing generalization
  - **Drill-Down**: Increasing levels of detail are revealed
  - **Pivot**: Cross tabulation is performed
  - **Slice** and **dice**: Performing projection operations on the dimensions
  - **Sorting**: Data is sorted by ordinal value
  - **Selection**: Data is available by value or range
  - **Derived attributes**: Attributes are computed by operations on stored derived values

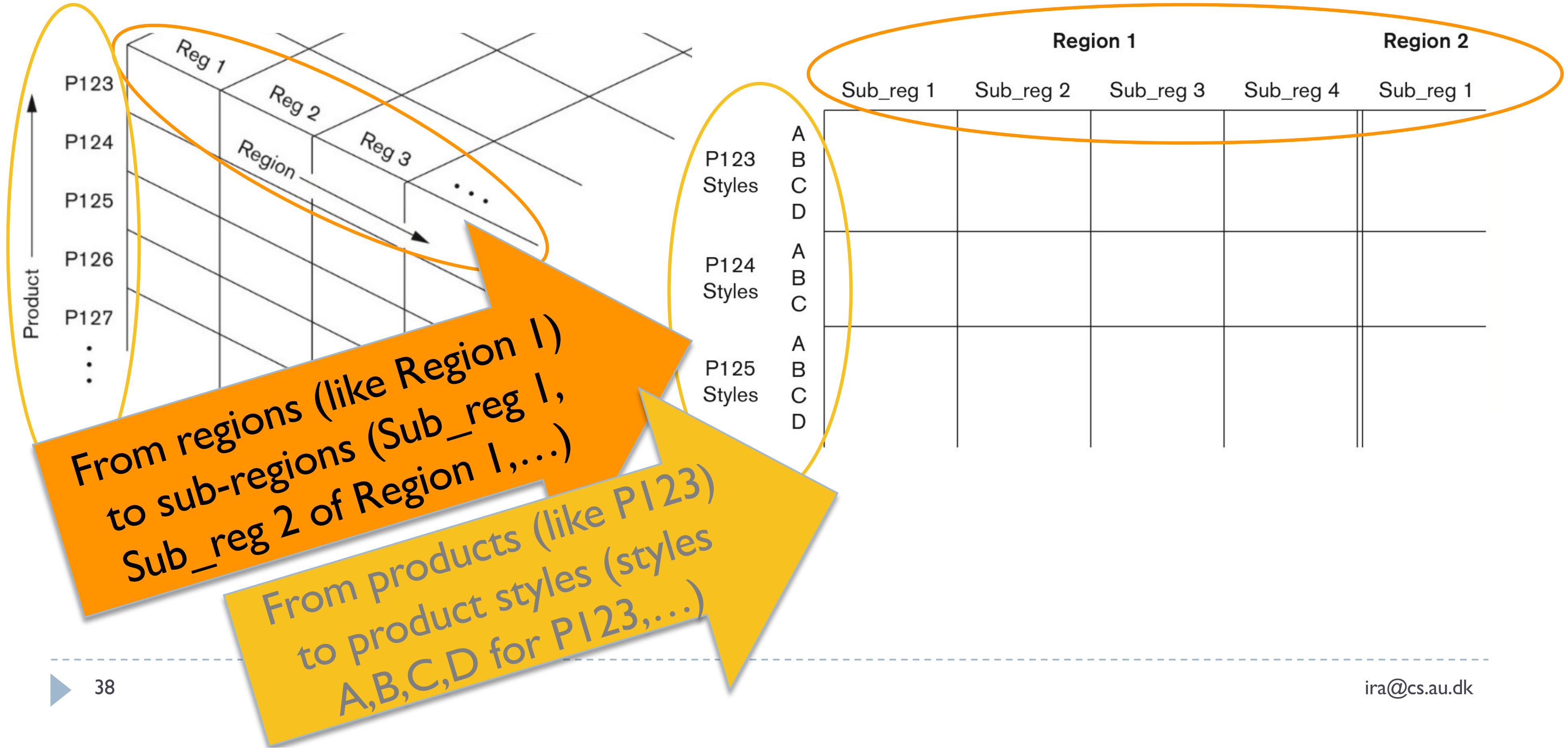
Roll-up

- Provide the means to "move up the hierarchy"
  - E.g. aggregate from smaller to larger regions or from more fine grained to more coarse product categories



Drill-down

- Opposite of roll-up
- Provides finer level information, e.g. from regions to sub-regions

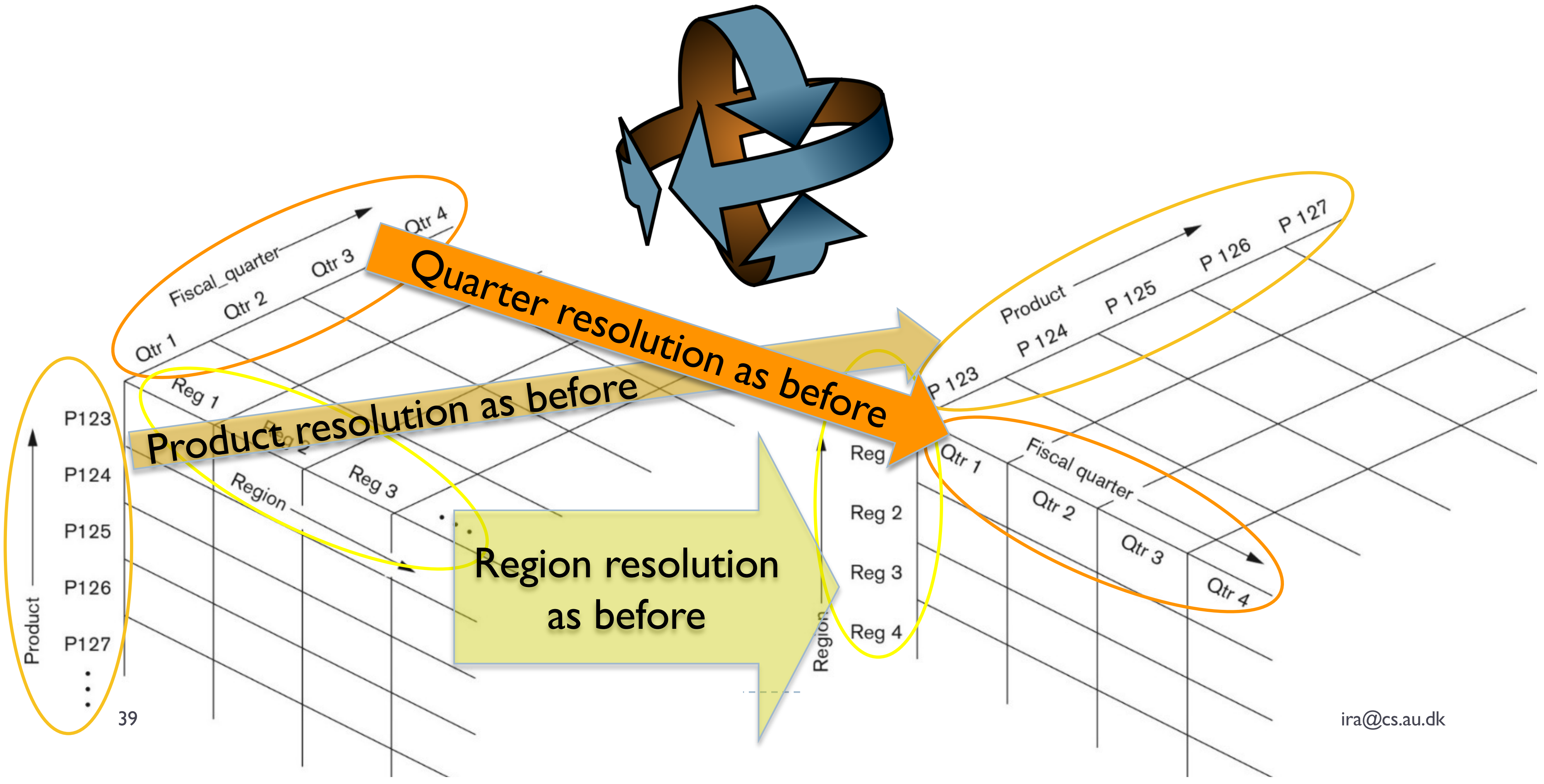


From regions (like Region 1) to sub-regions (Sub_reg 1, Sub_reg 2 of Region 1,…)

From products (like P123) to product styles (styles A,B,C,D for P123,…)

Pivoting (rotating)

- Same data, same resolution of the dimensions, but rotated
- Allows for ”reading” along a different ”axis”

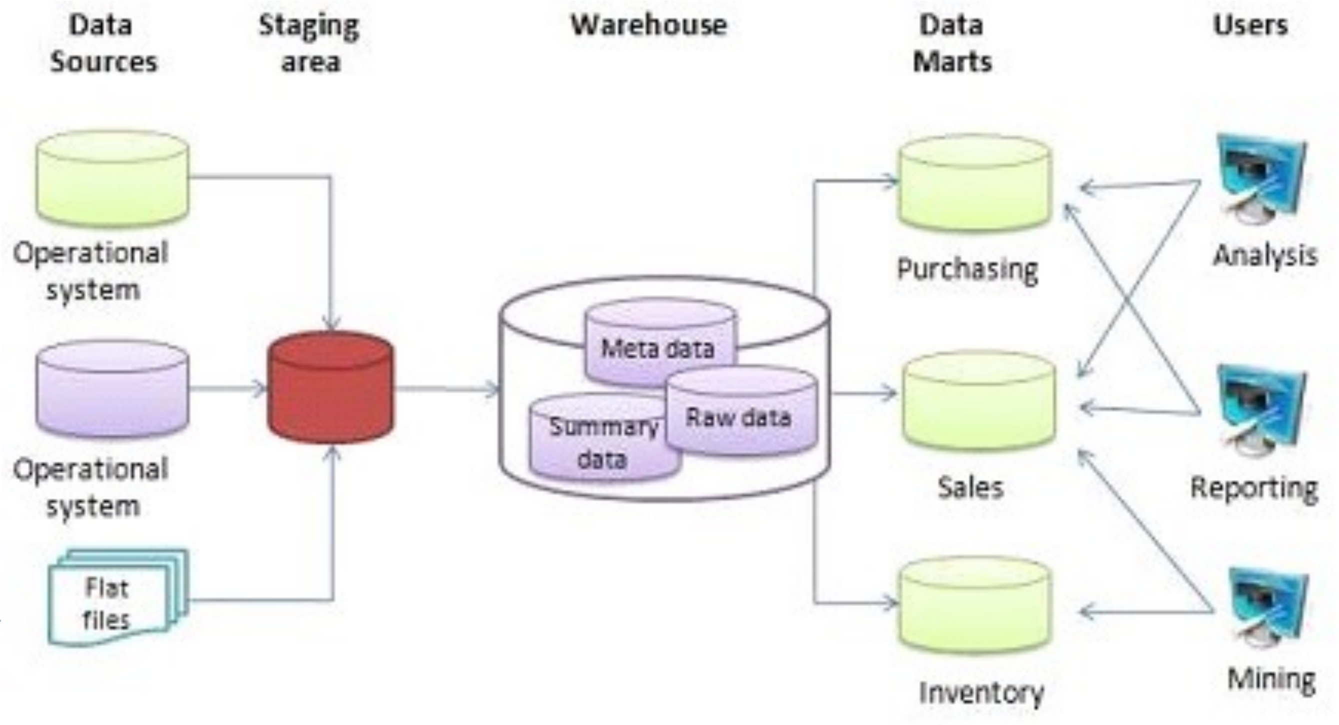


Difficulties of implementing Data Warehouses

- Lead time is huge in building a data warehouse
  - Potentially takes years to build and efficiently maintain a data warehouse
- Both quality and consistency of data are major concerns
- Revising the usage projections regularly to meet the current requirements
  - The data warehouse should be designed to accommodate addition and attrition of data sources without major redesign
- Administration of data warehouse requires different skills than are needed for a traditional database

Classification of Data Warehouses

- Data Warehouses generally an order of magnitude larger than source databases
  - Enterprise-wide data warehouses
    - Huge projects requiring massive investment of time and resources
  - Virtual data warehouses
    - Provide views of operational databases, but materialized for efficient access
  - Data marts
    - Targeted to subset of organization, such as a department, more tightly focused



OLAP and OLTP revisited

VOTE

I want OLAP, but I can only have OLTP, so I'll take a DB that's…

1. Denormalized and distributed
2. Normalized and distributed
3. Denormalized and centralized
4. Normalized and centralized

Comparison with Traditional Databases

- Data warehouses mainly optimized for appropriate data access
  - Traditional databases transactional and optimized for both access mechanisms and integrity assurance measures
- Data warehouses emphasize historical data as main purpose is to support (trend) analysis
- Compared with transactional databases, data warehouses are nonvolatile
- Transactional databases
  - transaction is the mechanism change to the database
- Information in data warehouse
  - relatively coarse grained
  - refresh policy is carefully chosen, usually incremental

Intended learning outcomes

- Be able to
  - Describe the main differences between data warehouses and transactional databases
  - Be able to use characterize main aspects of basic data warehouse design steps

What was this all about?

Guidelines for your own review of today's session

- As opposed to a database, a data warehouse is…
  - They are still similar in that…
- The main steps in building a data warehouse are…
  - The most challenging process ETL is…
- The typically used schemas in data warehouses are…
  - Their main difference is in…
  - We can express this information in SQL as follows…
- Views are similar to a data warehouse in that…
  - They differ with respect to…
- We navigate a data warehouse with the following functionalities…