Comp 5311 Database Management Systems

6. Advanced Topics in Databases

DB Topics

Data Warehouses and OLAP

Data Streams

Keyword Search in Databases

Spatial/Spatio-temporal Databases

Time Series

Skylines and Top-k Queries

AI and DB

Other Topics

DATA WAREHOUSES, OLAP, BIG DATA ANALYTICS

- *On-Line Transaction Processing* (OLTP) Systems manipulate operational data, necessary for day-to-day operations. Most existing database systems are in this category.
- *On-Line Analytical Processing* (OLAP) Systems support specific types of queries (based on group-bys and aggregation operators) useful for decision making.

Why OLTP is not sufficient for Decision Making

Lets say that Welcome supermarket uses a relational database to keep track of sales in all of stores simultaneously

SALES table			
product id	**store id**	**quantity sold**	**date/time of sale**
567	17	1	1997-10-22 09:35:14
219	16	4	1997-10-22 09:35:14
219	17	1	1997-10-22 09:35:17
...			

Example (cont.)

PRODUCTS table			
Prod. id	**product name**	**product category**	**Manufact. id**
567	Colgate Gel Pump 6.4 oz.	**toothpaste**	68
219	Diet Coke 12 oz. can	soda	5
...			

STORES table			
store id	**city id**	**store location**	**phone number**
16	34	510 Main Street	415-555-1212
17	58	13 Maple Avenue	914-555-1212

CITIES table			
id	**name**	**state**	**Popul.**
34	San Francisco	California	700,000
58	East Fishkill	New York	**30,000**

Example (cont.)

- An executive, asks "I noticed that there was a Colgate promotion recently, directed at people who live in towns with population < 100,000. How much Colgate toothpaste did we sell in those towns yesterday? And how much on the same day a month ago?"

```
select sum(sales.quantity_sold)
from sales, products, stores, cities
where products.manufacturer_id = 68   -- restrict to Colgate-
and products.product_category = 'toothpaste '
and cities.population < 100000
and sales.datetime_of_sale::date = 'yesterday'::date
and sales.product_id = products.product_id
and sales.store_id = stores.store_id
and stores.city_id = cities.city_id
```

Example (cont.)

- You have to do a 4-way JOIN of some large tables. Moreover, these tables are being updated as the query is executed.
- Need for a separate DB system (i.e., a Data Warehouse) to support queries like the previous one.
- The Warehouse can be tailor-made for specific types of queries: if you know that the toothpaste query will occur every day then you can denormalize the data model.

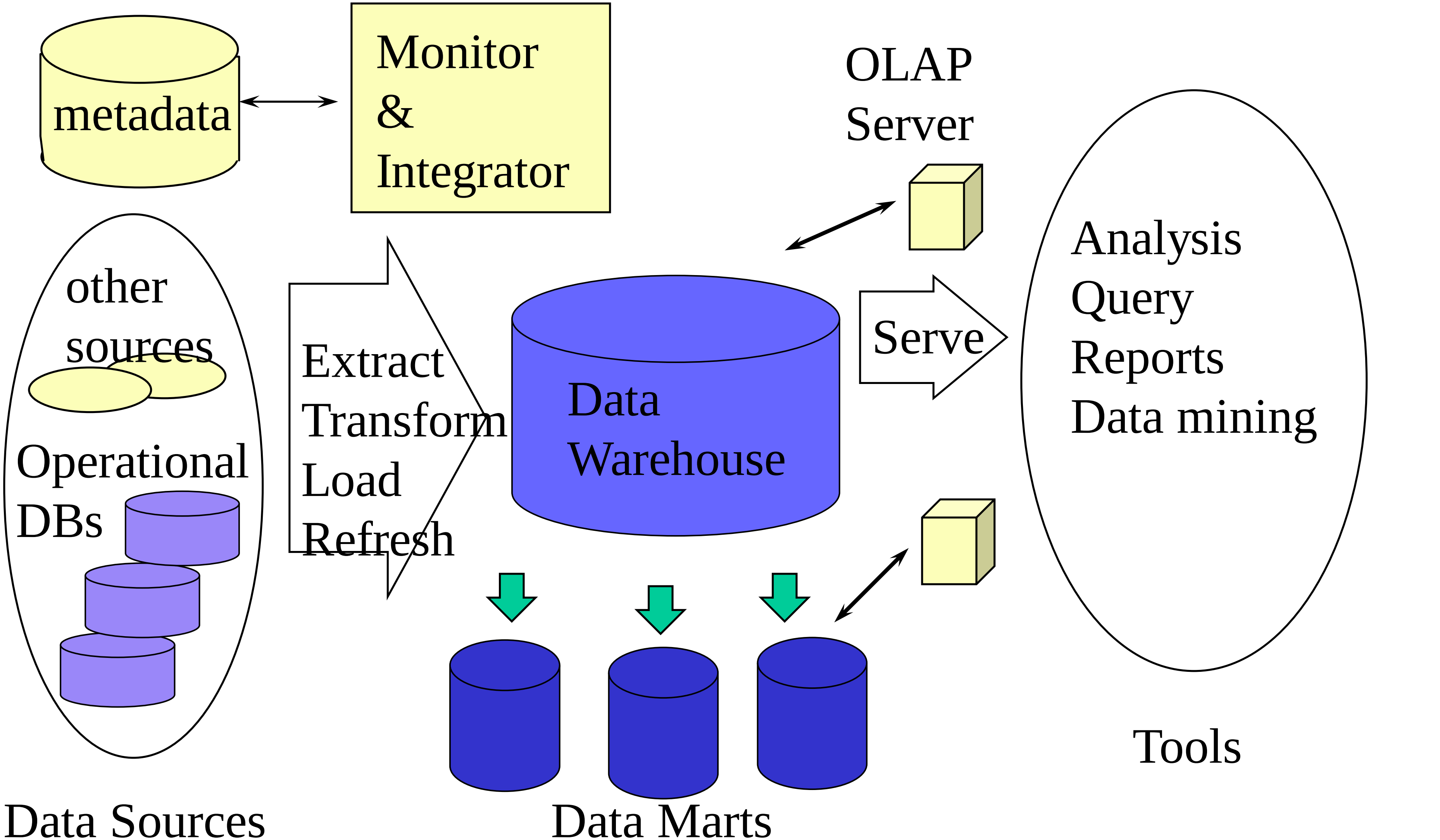
Example (cont.)

- Suppose Welcome acquires ParknShop which is using a different set of OLTP data models and a different brand of RDBMS to support them. But you want to run the toothpaste queries for both divisions.
- Solution: Also copy data from the ParknShop Database into the Welcome Data Warehouse (data integration).
- One of the more important functions of a data warehouse in a company that has disparate computing systems is to provide a view for management as though the company were in fact integrated.

Motivation

- In most organizations, data about specific parts of business is there - lots and lots of data - in some form.
- Data is available but *not information -- and not the right information at the right time.*
- To bring together information from multiple sources as to provide a consistent database source for decision support queries.
- To off-load decision support applications from the on-line transaction system.

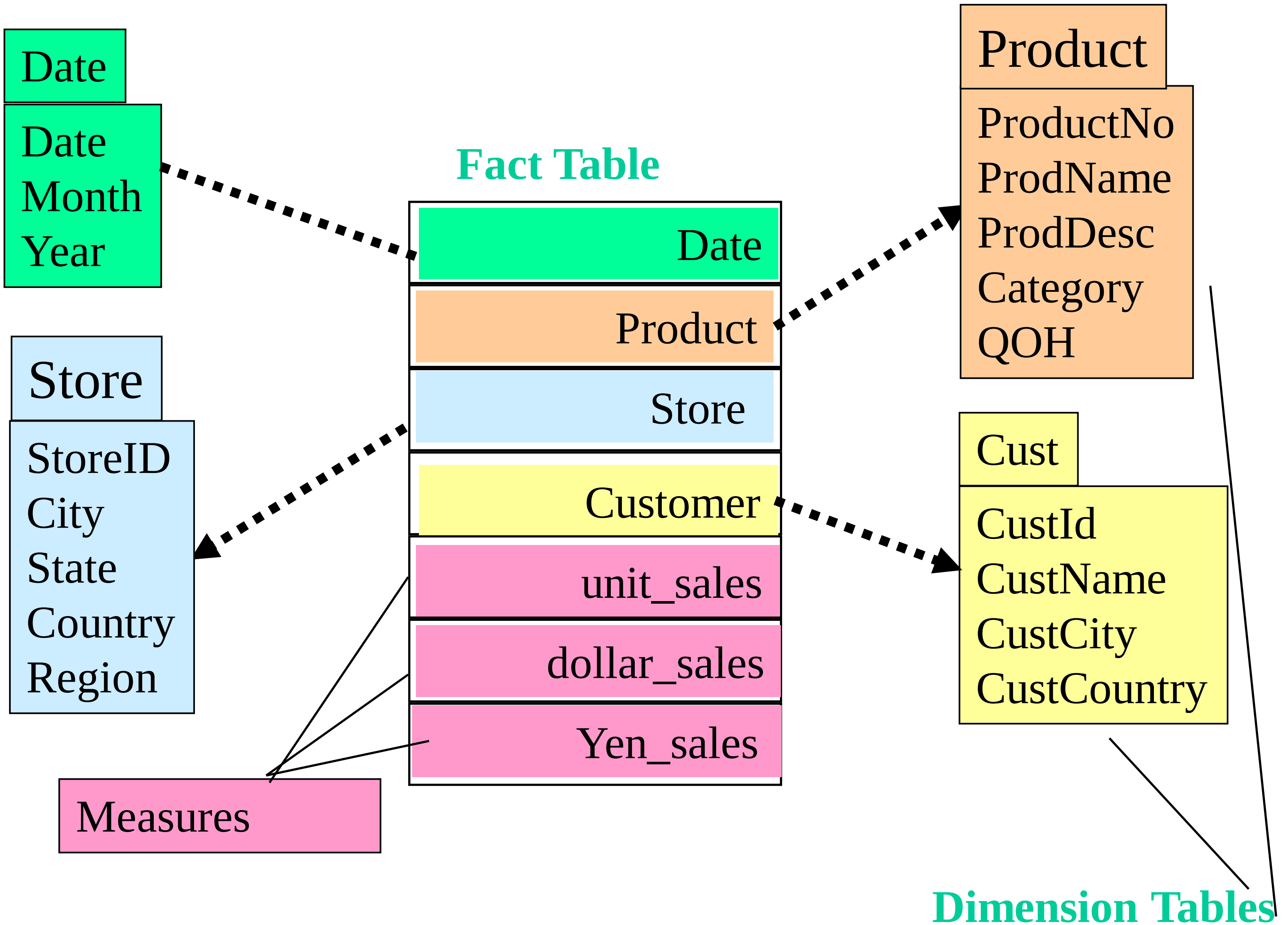
Multitiered Architecture



Loading Data to the Warehouse

- The warehouse must clean data, since operational data from multiple sources are often dirty: inconsistent field lengths, missing entries, violation of integrity constraints.
- Loading the warehouse includes some other processing tasks: checking integrity constraints, sorting, summarizing, build indexes, etc.
- Refreshing a warehouse means propagating updates on source data to the data stored in the warehouse

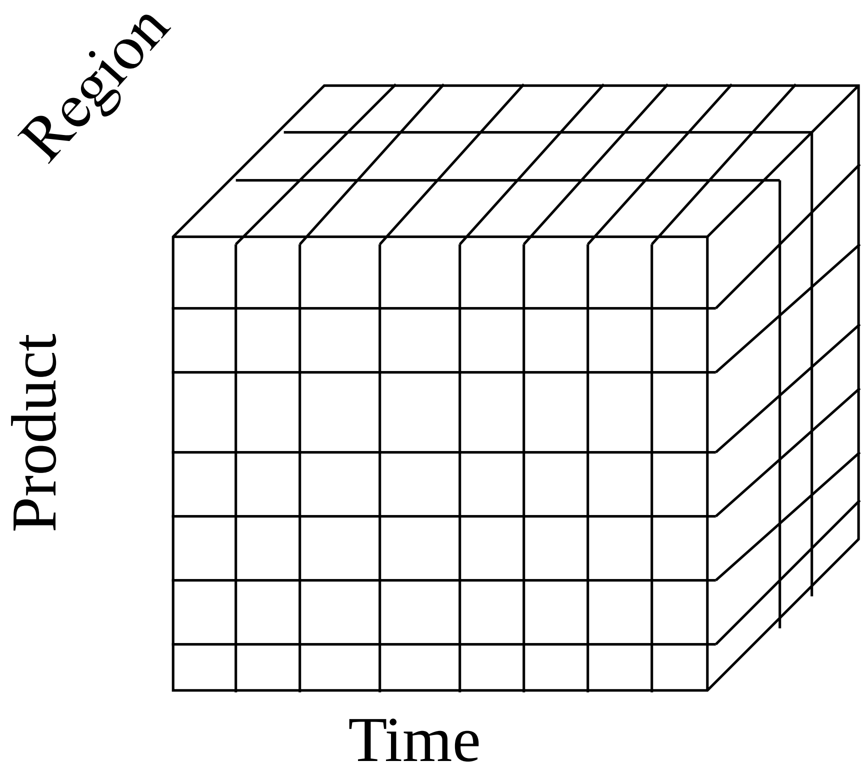
Conceptual Modeling - Star Schema

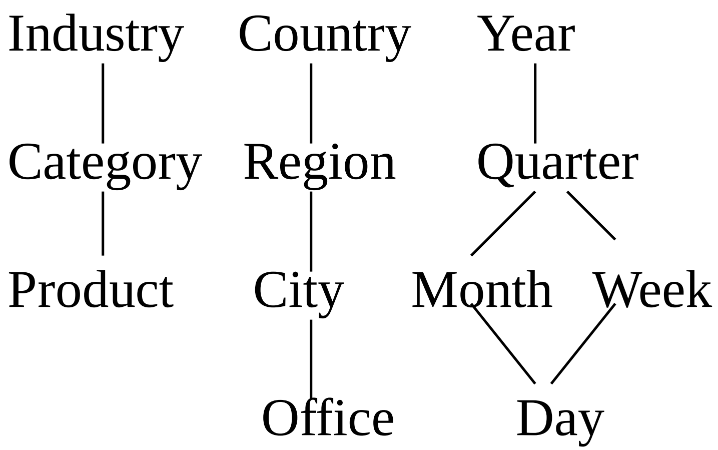


Multidimensional View of Data

- Sales volume (measure) as a function of product, time, and geography (dimensions).

Dimensions: Product, Region, Time
Hierarchical summarization paths

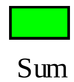




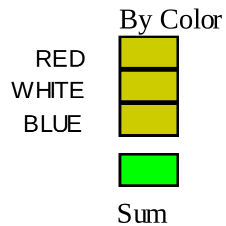
Data Cube = Fact table + All Group-bys

```
SELECT SUM(Sales)
FROM Sales
```

Aggregate



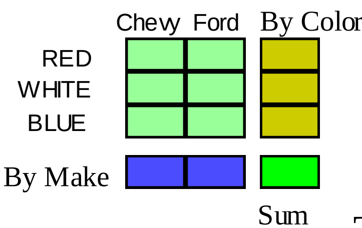
Group By (with total)



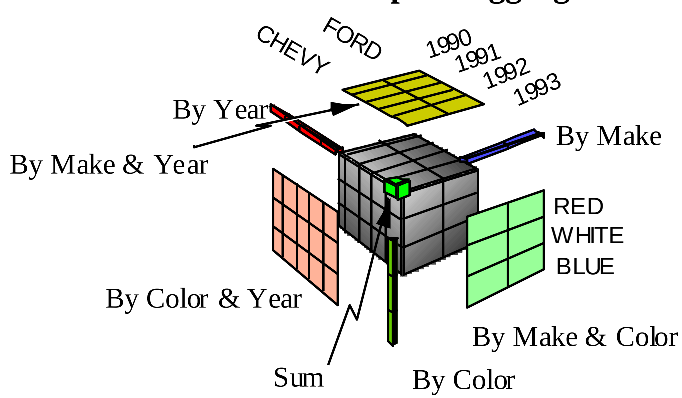

```
SELECT Color, SUM(Sales)
FROM Sales
 GROUP BY     Color
```

```
SELECT Color, Model,  SUM(Sales)
FROM Sales
 GROUP BY     Color, Model
```

Cross Tab



The Data Cube and The Sub-Space Aggregates

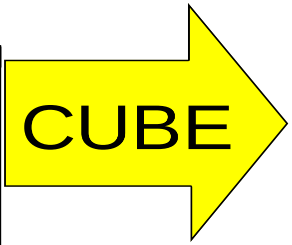


Cube Operation

```
SELECT Model, Year, Color, SUM(sales)
FROM Sales
GROUP BY CUBE(Model, Year, Color);
```

SALES

Model	Year	Color	Sales
Chevy	1990	red	5
Chevy	1990	white	87
Chevy	1990	blue	62
Chevy	1991	red	54
Chevy	1991	white	95
Chevy	1991	blue	49
Chevy	1992	red	31
Chevy	1992	white	54
Chevy	1992	blue	71
Ford	1990	red	64
Ford	1990	white	62
Ford	1990	blue	63
Ford	1991	red	52
Ford	1991	white	9
Ford	1991	blue	55
Ford	1992	red	27
Ford	1992	white	62
Ford	1992	blue	39



DATA CUBE

Model	Year	Color	Sales
Chevy	1990	blue	62
Chevy	1990	red	5
Chevy	1990	white	95
Chevy	1990	ALL	154
Chevy	1991	blue	49
Chevy	1991	red	54
Chevy	1991	white	95
Chevy	1991	ALL	198
Chevy	1992	blue	71
Chevy	1992	red	31
Chevy	1992	white	54
Chevy	1992	ALL	156
Chevy	ALL	blue	182
Chevy	ALL	red	90
Chevy	ALL	white	236
Chevy	ALL	ALL	508
Ford	1990	blue	63
Ford	1990	red	64
Ford	1990	white	62
Ford	1990	ALL	189
Ford	1991	blue	55
Ford	1991	red	52
Ford	1991	white	9
Ford	1991	ALL	116
Ford	1992	blue	39
Ford	1992	red	27
Ford	1992	white	62
Ford	1992	ALL	128
Ford	ALL	blue	157
Ford	ALL	red	143
Ford	ALL	white	133
Ford	ALL	ALL	433
ALL	1990	blue	125
ALL	1990	red	69
ALL	1990	white	149
ALL	1990	ALL	343
ALL	1991	blue	106
ALL	1991	red	104
ALL	1991	white	110
ALL	1991	ALL	314
ALL	1992	blue	110
ALL	1992	red	58
ALL	1992	white	116
ALL	1992	ALL	284
ALL	ALL	blue	339
ALL	ALL	red	233
ALL	ALL	white	369
ALL	ALL	ALL	941

DATA STREAMS

- Data streams differ from conventional DMBS:
  - Records arrive online
  - System has no control over arrival order
  - Data streams are potentially unbounded in size
  - Once a record from a data stream has been processed, it is discarded or archived. It cannot be retrieved easily because memory is small relative to the size of data streams
- Continuous queries
  - Snapshot queries in conventional databases
    - Evaluated once over a point-in-time snapshot of data set
  - Continuous queries in data streams
    - Evaluated continuously as data streams continue to arrive
    - May be stored and updated as new data arrives, or may produce data streams themselves

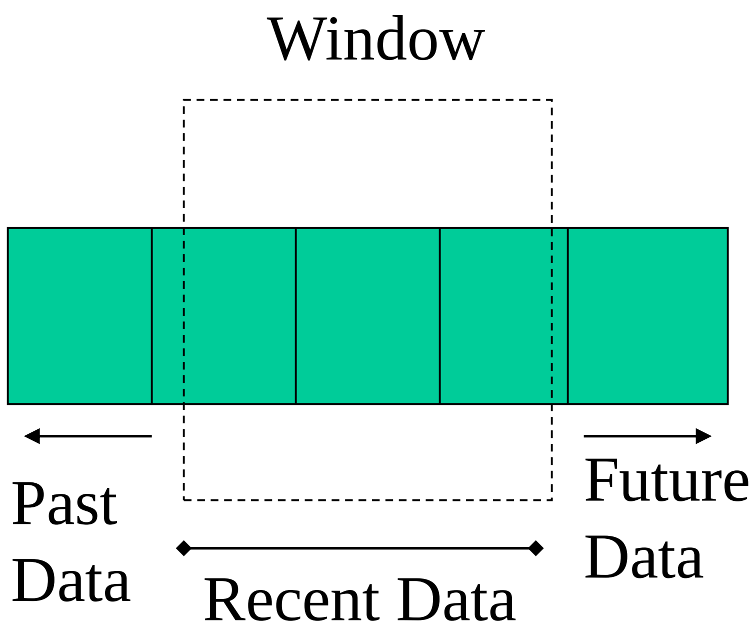
Motivating Examples

- Financial system receiving stock values.
  - "sell the stock when the value drops below $10".
- Modern security applications.
  - "detect potential attacks to the network"
- Clickstream monitoring to enable applications such as personalization, and load-balancing. (e.g., Yahoo)
- Sensor monitoring
  - "identify traffic congestions in road networks using sensors monitoring traffic"

Finite Streams

- Finite Streams are bounded (i.e., at some point all tuples arrive)
- Unlike conventional databases, processing takes place in main memory, without all the data available in advance
- Conventional join algorithms require one input (BNL, index nested loop) or both inputs (sort merge and hash join) in advance
- Adapted versions of the algorithms for streams:
  - must produce the first results immediately after the arrival of the first tuples
  - must keep a "constant" output rate
  - must utilize the available main memory

Infinite Streams - Sliding Windows



- Infinite Streams: data are NOT bounded (they arrive for ever).
- Evaluate query over sliding window of recent data from streams
- Attractive Properties
  - Well-defined and understood
  - Emphasizes recent data, which in many real-world applications is more important than old data

Sliding Windows - Joins

- Two tuples can be joined only if they fall in the same sliding window (i.e., there time difference is within the window).
- General framework for joining streams *A* and *B*. Tuples arrive in chronological order. System maintains the list of tuples S_A and S_B that have arrived and not expired yet.
  - An incoming tuple *t* from input stream *A* first *purges* tuples from S_B whose timestamp is earlier than *t.ts*-*w*.
  - Then, it *probes* S_B and joins with its tuples.
  - Finally, *t* is *inserted* into S_A.
- Once a join result is generated, it must also be assigned a timestamp, since it may constitute an input for a subsequent operator.
- Output tuples must be generated in the order of their timestamps

Data Streams – Other Issues

- Approximate Queries – due to limited amount of memory, it may not be possible to produce exact answers
  - Sketches
  - Random sampling
  - Histograms
  - Wavelets
- Query optimization
  - How to optimize continuous queries
  - How to migrate plans

RELATIONAL KEYWORD SEARCH

KEYWORD SEARCH (KWS)

Very Easy

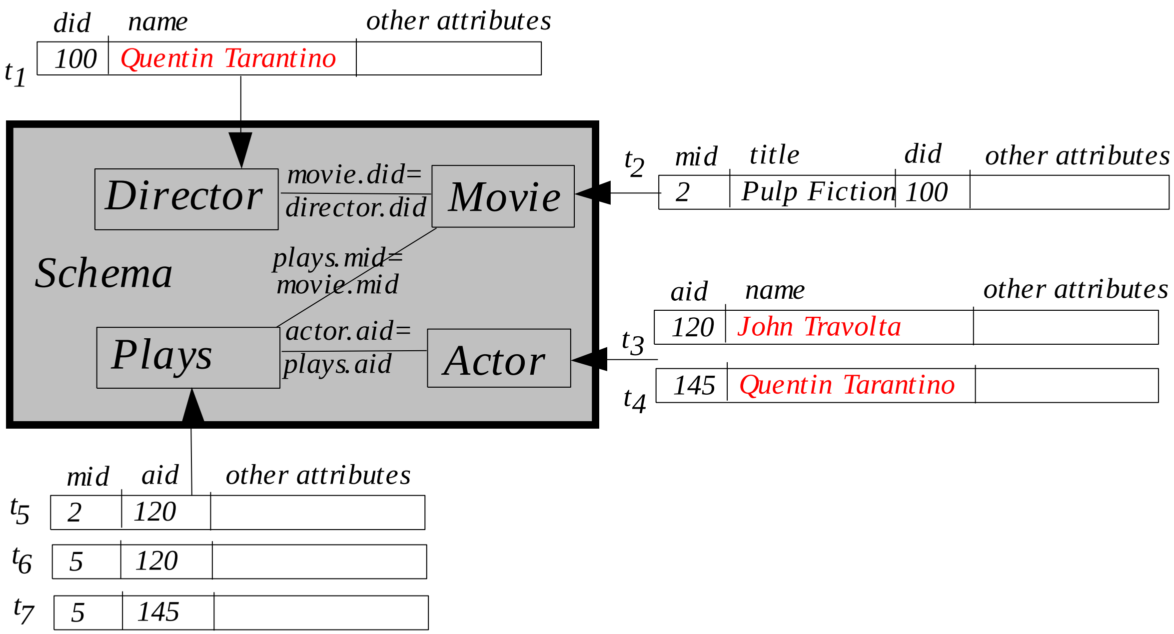
- No language to learn

Ubiquitous

- Web Search
  - Millions of users
  - Millions of queries

Now applied to databases…

Example of KWS



What is the query {Tarantino, Travolta} supposed to compute?

- *t1* JOIN *t2* JOIN *t5* JOIN *t3*: there is a movie (Pulp Fiction), which was directed by Tarantino and features Travolta
- *t3* JOIN *t6* JOIN *t7* JOIN *t4* : there is movie (*mid*=5) that includes both Tarantino and Travolta as actors

Equivalent SQL Expressions

```
SELECT *
FROM Director D, Movie M,
      Plays P, Actor A
WHERE D.name=Tarantino, A.name=Travolta
and D.did=M.did and P.mid=M.mid
and A.aid=m.aid
```

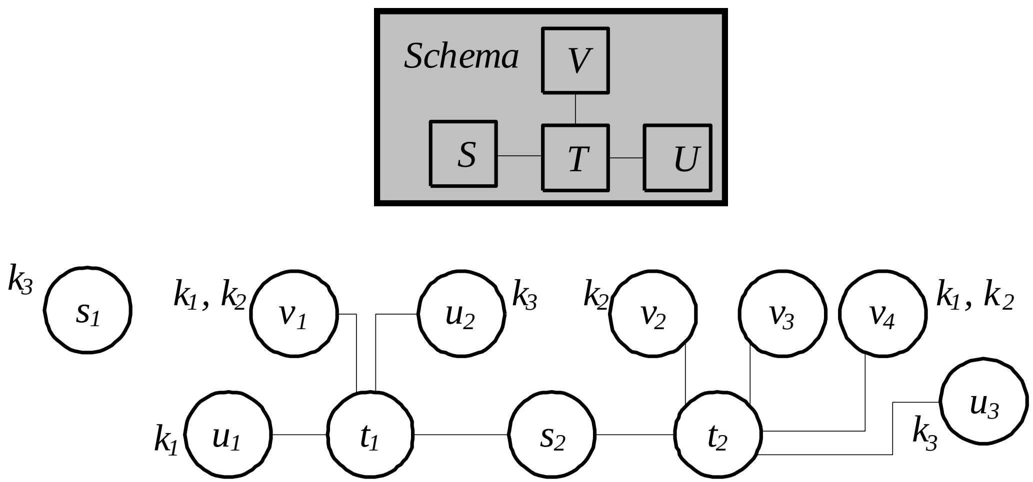
```
SELECT *
FROM Actor A1, Actor A2, Plays P1, Plays P2,
WHERE A1.name=Tarantino, A2.name=Travolta
and A1.aid=P1.aid and A2.aid=P2.aid and
P1.mid=P2.mid
```

These are only the statements that actually output results. Many more SQL queries have to be issued, in order to cover every possible interaction, e.g. a movie starring Tarantino that was directed by Travolta.

R-KWS allows querying for terms in unknown locations (tables/attributes). A query can be issued without knowledge of tables, their attributes, or join conditions.

Database as a Graph

- Every Database can be modeled as a graph*:
- Nodes
  - Represent tuples
- Edges
  - Connect joining tuples



Data Graph

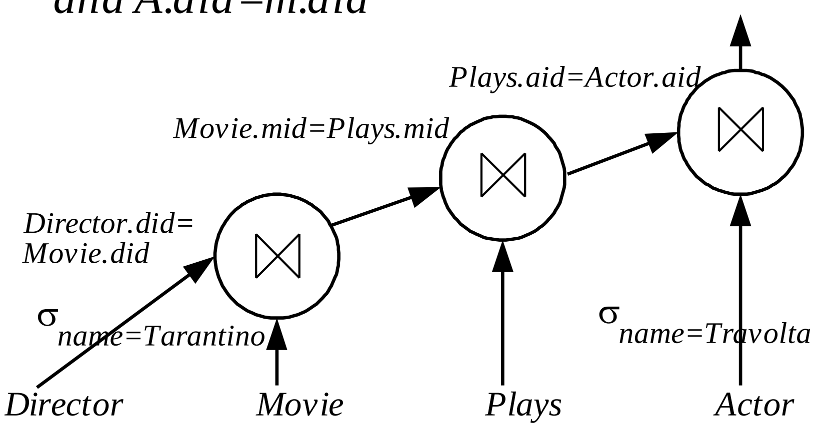
Graph-Based Query Processing

- Graph based systems such as *Banks* and *DBSurfer* maintain the data graph in main memory.
- Given a query, an inverted index identifies all tuples that contain at least one keyword.
- Each such tuple initiates a graph traversal.
- Whenever a node is reached by all keywords, a result is constructed by following the reverse paths to the keyword occurrences.
- Duplicates are filtered in a second, post-processing step.

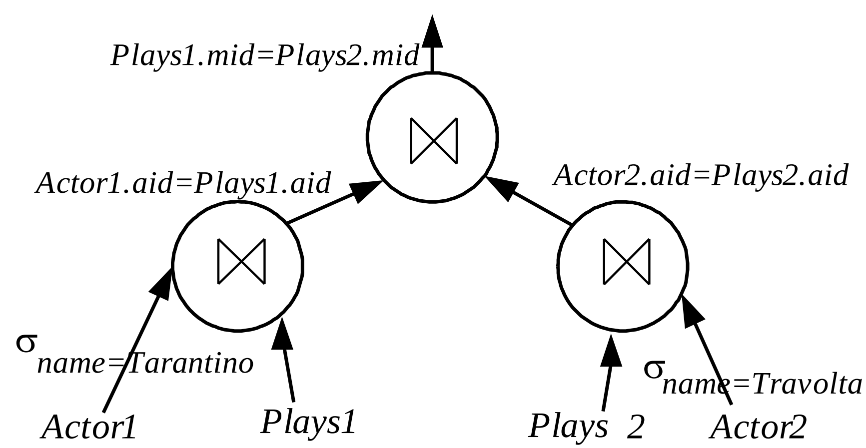
Operator-Based Query Processing

- Systems, such as *Discover*, *DBXplorer* and *Mragyati*, translate an R-KWS query into a series of SQL statements, which are executed directly on secondary storage, using the underlying DBMS.

*SELECT **
FROM Director D, Movie M,
Plays P, Actor A
WHERE D.name=Tarantino, A.name=Travolta
and D.did=M.did and P.mid=M.mid
and A.aid=m.aid



*SELECT **
FROM Actor A1, Actor A2, Plays P1, Plays P2,
WHERE A1.name=Tarantino, A2.name=Travolta
and A1.aid=P1.aid and A2.aid=P2.aid and
P1.mid=P2.mid



Database Keyword Search – Other Topics

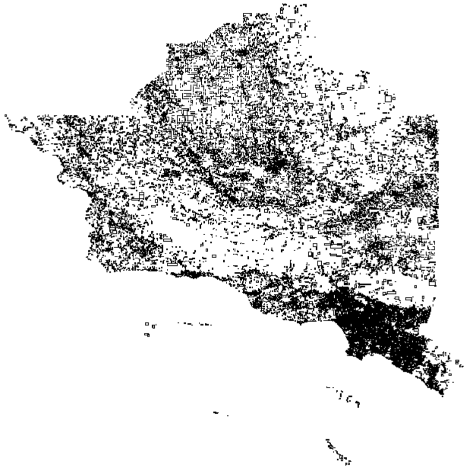
- Natural language to SQL
- Ranking – How to retrieve the top-k most interesting results
- Keyword search in multiple databases
  - How to select the top-k databases with the most promising results
- Continuous keyword search in streams
- Spatial keyword search

SPATIAL AND SPATIOTEMPORAL DATABASES

Spatial Database Systems manage large collections of static multidimensional objects with explicit knowledge about their extent and position in space (as opposed to *image databases*).

A *spatial object* contains (at least) one spatial attribute that describes its geometry and location

A *spatial relation* is an organized collection of spatial objects of the same entity (e.g. rivers, cities, road segments)



Road segments from an area in CA

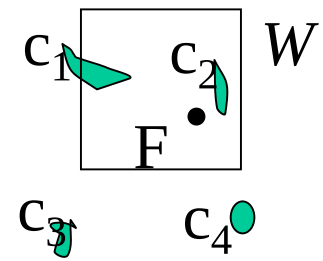
ID	Name	Type	Polyline
1	Sunset	avenue	(10023,1094), (9034,1567), (9020,1610)
2	H5	highway	(4240,5910), (4129,6012), (3813,6129), (3602,6129)
…	…	…	…

A spatial relation

Common Spatial Queries

- Range query (spatial selection, window query, zoom-in)

  e.g. find all cities that *intersect* window *W*
  Answer set: $\{c_1, c_2\}$

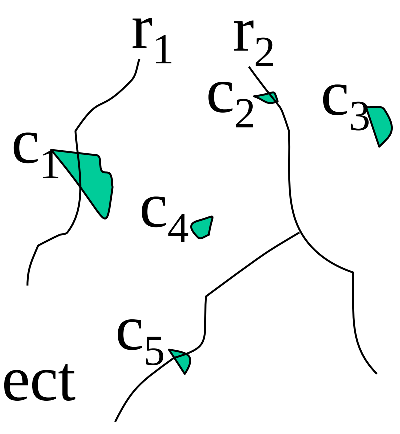


- Nearest neighbor query

  e.g. find the city closest to the F-spot
  Answer: c_2

- Spatial join

  e.g. find all pairs of cities and rivers that intersect
  Answer set: $\{(r_1,c_1), (r_2,c_2), (r_2,c_5)\}$

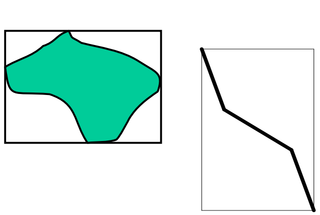


Two-step spatial query processing

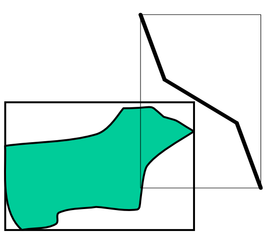
- A *spatial object* is usually approximated by its *minimum bounding rectangle* (MBR)
- The *spatial query* is then processed in two steps:

  1. *Filter step*: The MBR is tested against the query predicate
  2. *Refinement step*: The exact geometry of objects that pass the filter step is tested for qualification

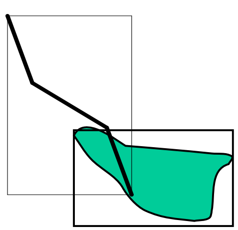
Examples:



filtered pair



non-qualifying pair that passes the filter step (false hit)

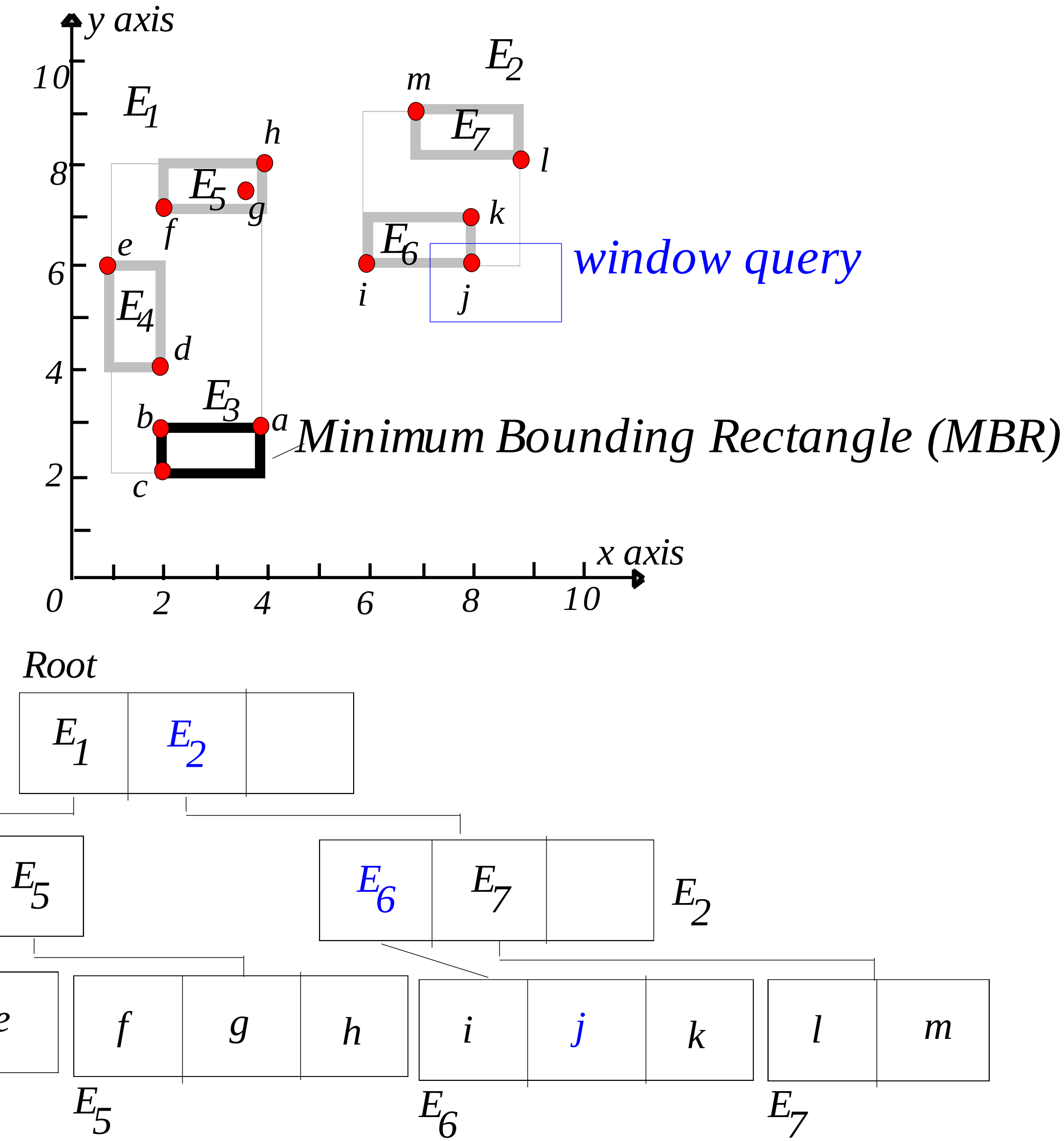


qualifying pair

Example R-tree – Range (Window) Query

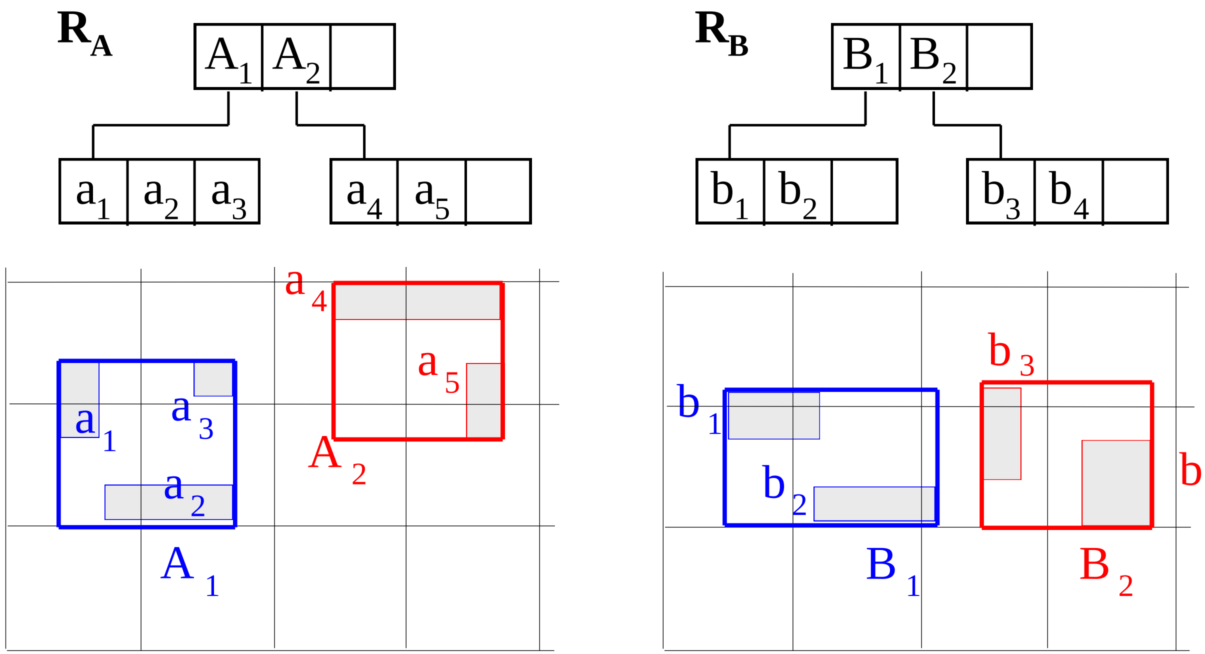
Node capacity depends on the page size

In practice in the order of 100



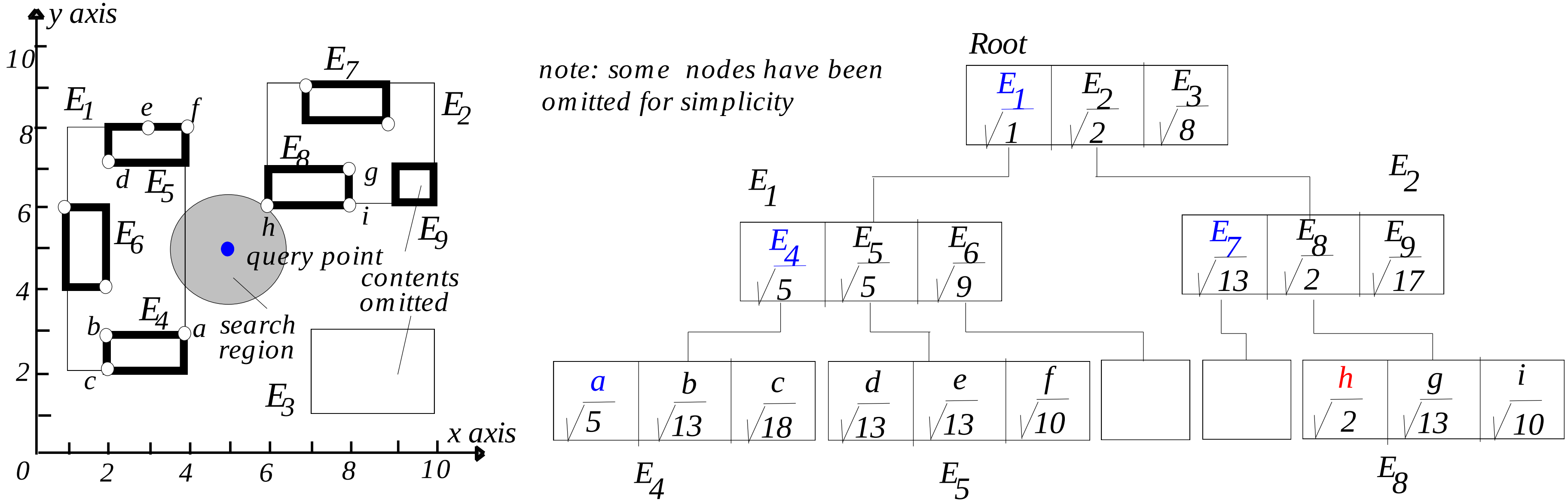
Spatial Joins

- A spatial join returns intersecting pairs of objects (from two data sets)
- The RJ join algorithm traverses both R-trees simultaneously, visiting only those branches that can lead to qualifying pairs.



Nearest Neighbor (NN) search with R-trees

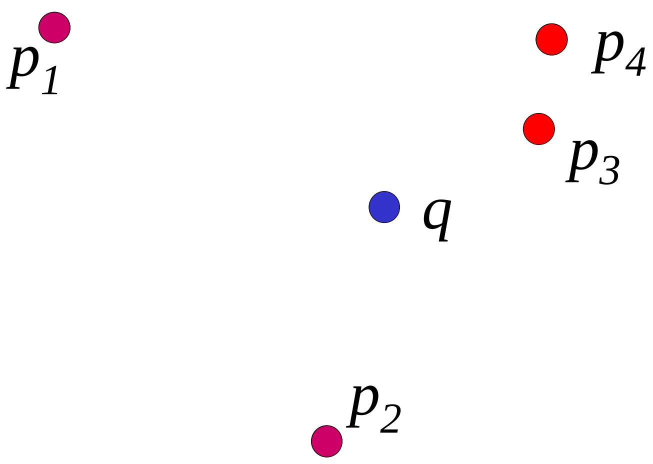
- *Depth-first traversal*



Reverse NN Queries

Monochromatic: given a multi-dimensional dataset P and a point q, find all the points $p \in P$ that have q as their nearest neighbor

Bichromatic: given a set Q of queries and a query point q, find the objects $p \in P$ that are closer to q than any other point of Q



RNN(q)=p_1, p_2

NN(q)= p_3

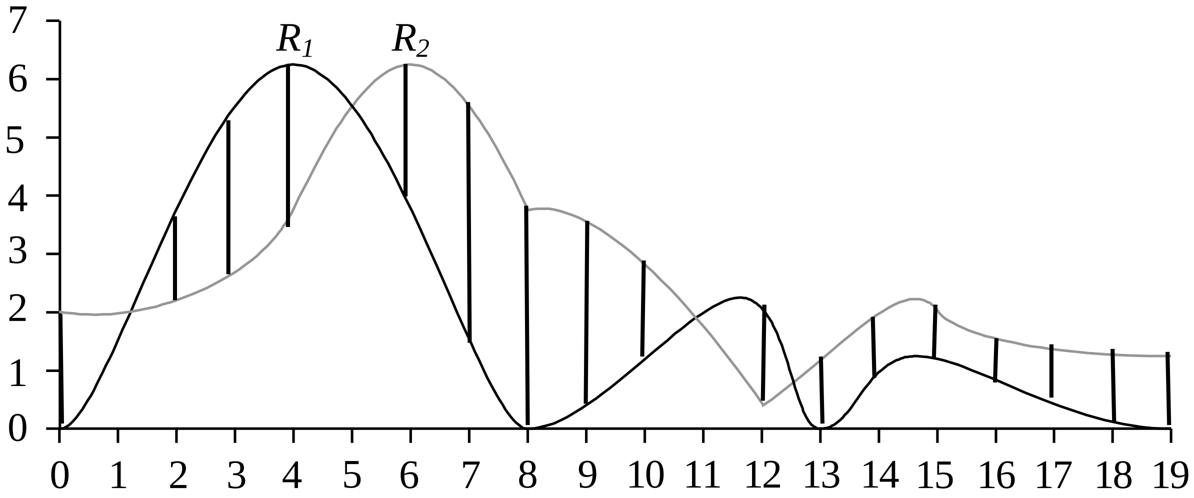
Spatial and Spatiotemporal DB – Other Issues

- Road networks
- Continuous monitoring of spatial queries
- Predictive indexing and query processing
- Indexing historical location data
- Spatiotemporal aggregation
- Alternative types of spatial queries
- Location privacy
- Learned Spatial Indexes

TIME SERIES DATABASES

- A *time series* or *data sequence R* consists of a stream of numbers ordered by time: $R = R[0], R[1], \ldots,$ where $R[0]$ corresponds to the value at timestamp 0, $R[1]$ to the value at timestamp 1 and so on.
- Time series are ubiquitous in several applications: stock market, image similarity, sensor networks etc.
- Queries: Similarity Search (find all stocks whose values in the last year are similar to a given stock).

Similarity Definition



- Difficult to define – depends on the application domain, user.
- A simple definition is based on Euclidean distances
- Does not account for translation, rotation etc.

Whole Sequence Matching

- Given a set of stored time series with the same length d, a query sequence Q with length d and a similarity threshold ε, a *whole matching* query returns the series that ε-match with Q.
- 3-step processing framework
  - *index building*: apply dimensionality reduction technique to convert d-dimensional sequences to points into an f-dimensional space. The resulting f-dimensional points are indexed by an R-tree
  - *index searching*: transform the query sequence Q to an f-dimensional point q. A range query centered at q with radius ε is performed on the R-tree to retrieve candidates results.
  - *post-processing* is performed on the candidates to get actual result.

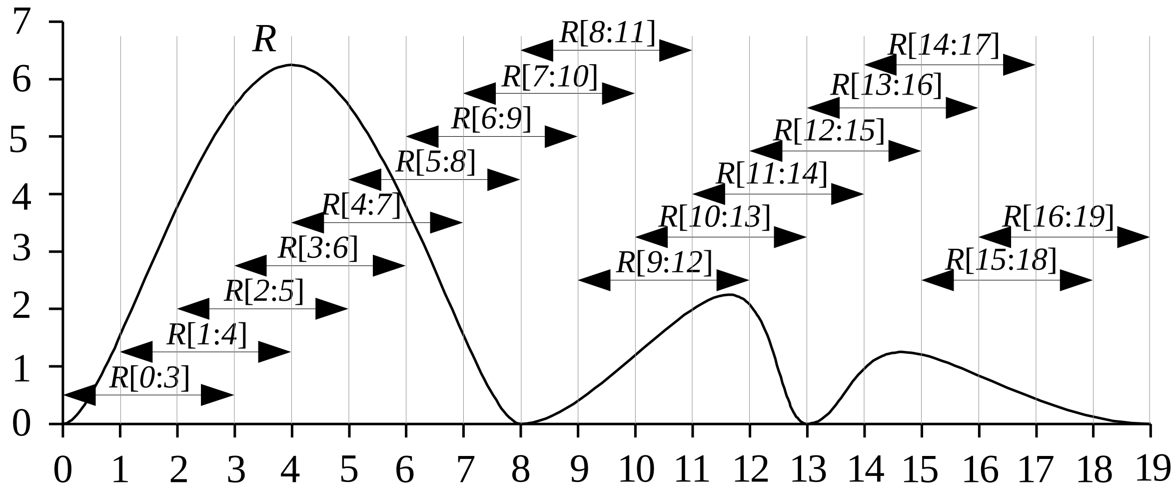
Whole Sequence Matching - Assumptions

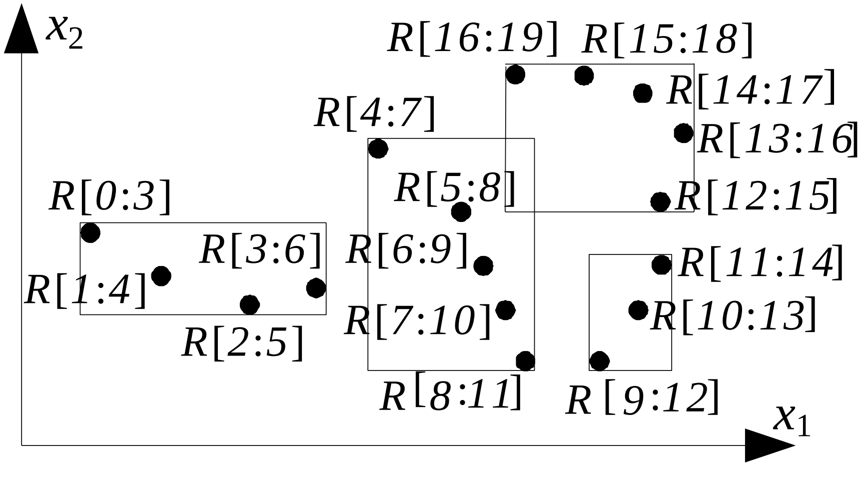
- All data base sequences and query sequence should have the same length
- The dimensionality reduction technique should be distance-preserving: i.e., the distance in the low dimensional space should be smaller or equal to the distance in high dimensions

Sub-Sequence Matching

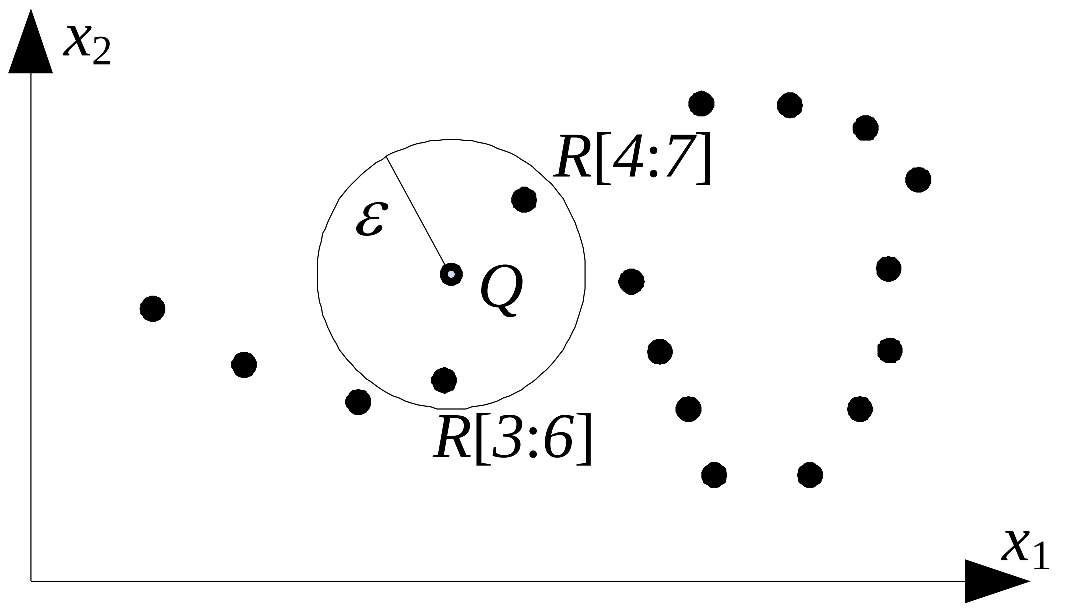
Given a data sequence $R = R[0], \ldots, R[m-1]$, a query sequence $Q = Q[0], \ldots, Q[d-1]$ ($m \geq d$) and a similarity threshold ε, a sub-sequence matching query retrieves all the subsequences $R' = R[i : i+d-1]$ ($0 \leq i \leq m-d$), such that $dist(Q, R') \leq \varepsilon$.

Index Building for Sub-Sequence Matching





Query processing - Query length w (=4)



Time Series – Other Issues

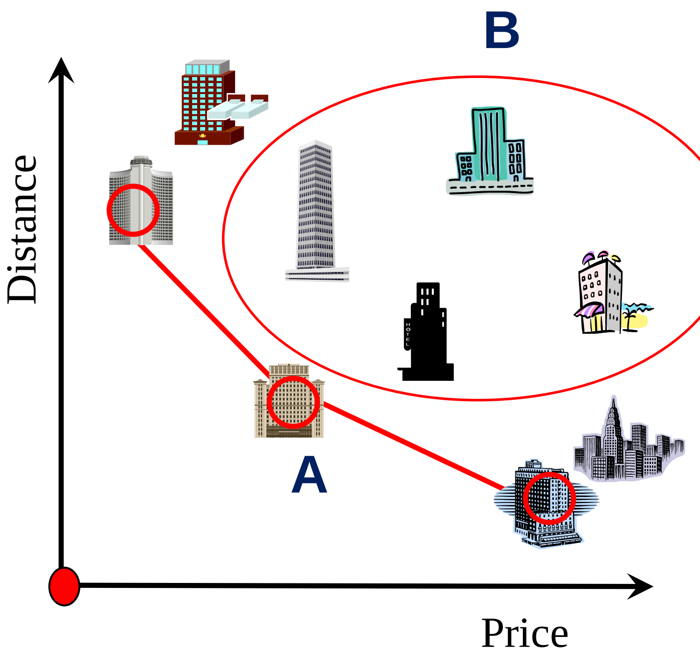
- Distance definitions
  - Dynamic Time Warping
  - Application-dependent definitions
- Dimensionality reduction techniques
  - Discrete Fourier Transform
  - Wavelets
  - Linear Segments
- Alternative problems
  - Outlier detection
  - Streaming time series

SKYLINE AND TOP-*K* QUERIES

- Which buildings can we see?
  - Higher or nearer

Skyline Example

Find a cheap hotel that is close to beach.



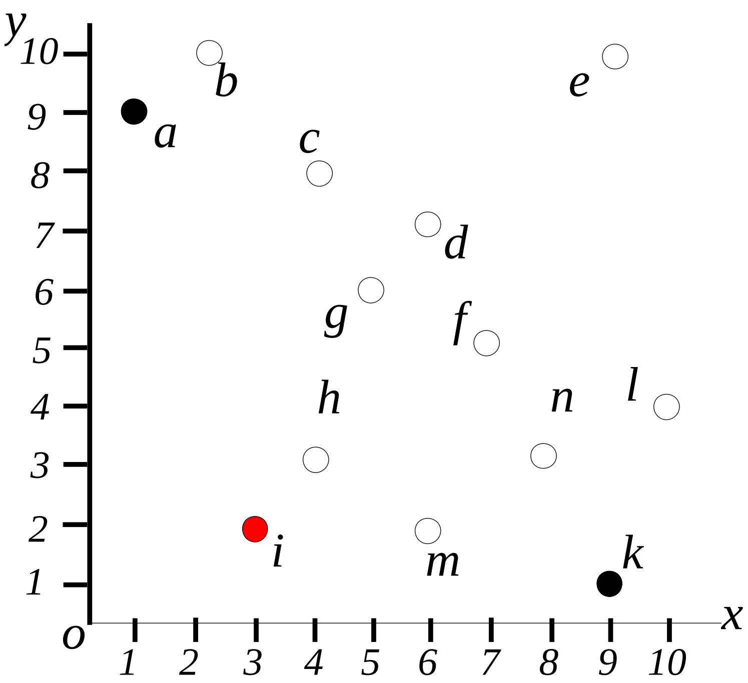
A *dominates* **B**.

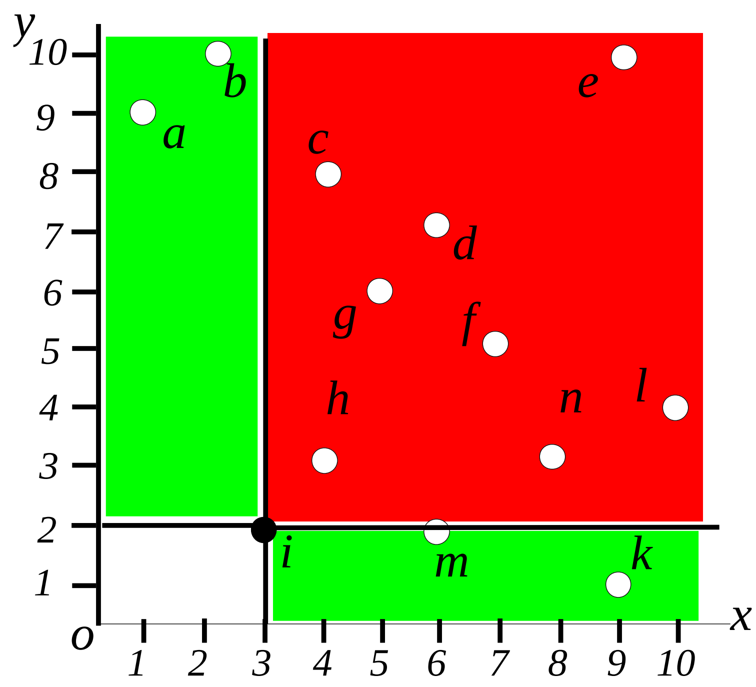
→ A(dist) ≤ B(dist) and A(price) ≤ B(price)

Skyline is a set of objects not dominated by any other objects.

NN algorithm

NN uses the results of nearest neighbor search to partition the data universe recursively.

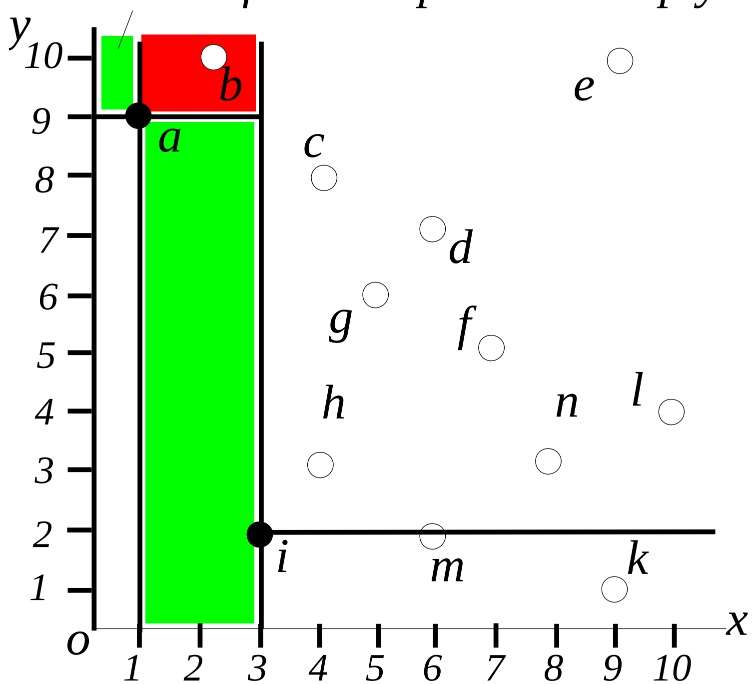


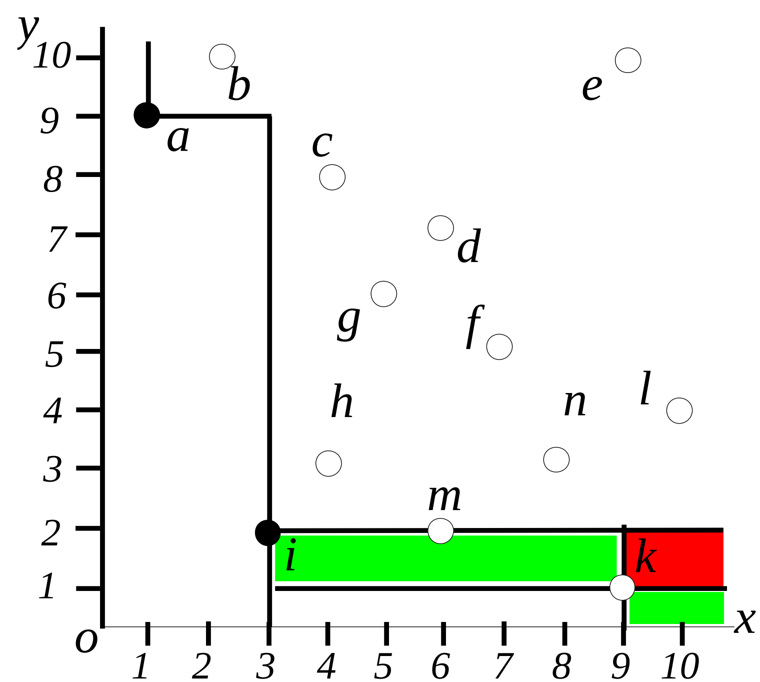


NN algorithm (cont)

- NN uses the results of nearest neighbor search to partition the data universe recursively.

note: another query is necessary to confirm this partition empty





Top-*k* queries

Top-*k* query: Given a scoring function *f*, report the *k* tuples in a dataset with the highest scores.

fundid	1	2	3	4	5	**6**	7	8	9	10	**11**	**12**
growth	0.2	0.1	0.3	0.2	0.3	**0.5**	0.4	0.6	0.7	0.6	**0.7**	**0.7**
stability	0.2	0.5	0.3	0.9	0.8	**0.7**	0.3	0.1	0.2	0.5	**0.6**	**0.5**

Preference function $f(t)=w1 \cdot t.growth+w2 \cdot t.stability$

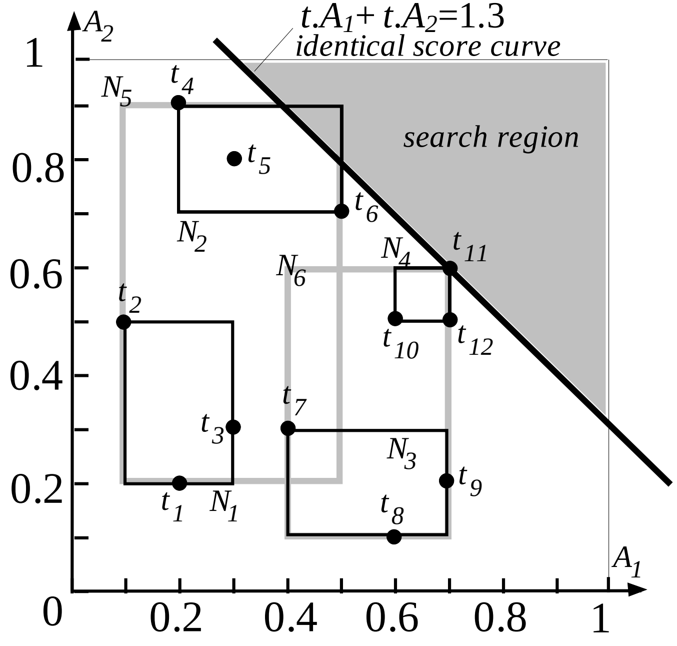
where w1 and w2 are specified by a user to indicate her/his priorities on the two attributes.

- *If w1=0.1, w2=0.9 (stability is favored), the top 3 funds have ids 4, 5, 6 since their scores (0.83, 0.75, 0.68, respectively) are the highest.*
- *If w1=0.5, w2=0.5 (both attributes are equally important), the ids of the best 3 funds become* ***11, 6, 12.***

Top-*k* query Processing

Query processing techniques

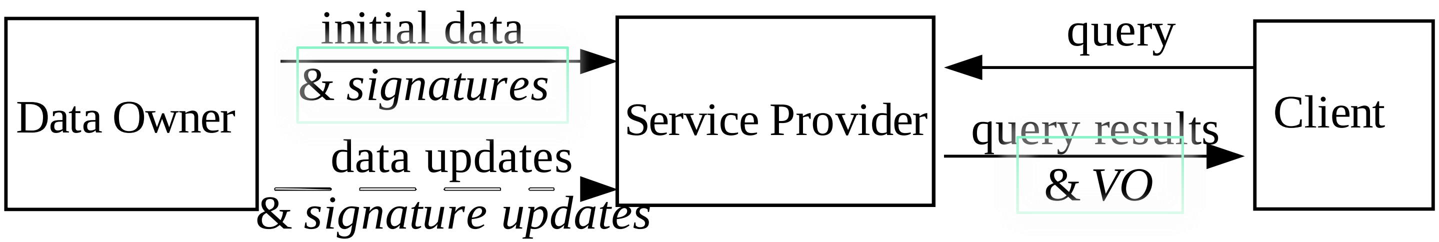
- Based on pre-processing (i.e., generation of views in advance)
- On-line (no preprocessing)



DATABASE OUTSOURCING AND QUERY AUTHENTICATION

- Advantages
  - The data owner does not need the hardware / software / personnel to run a DBMS
  - The service provider achieves economies of scale
  - The client enjoys better quality of service
- A main challenge
  - The service provider is not trusted, and may return incorrect query results (e.g., to favor the competition)

Framework



- The owner signs its data with a digital signature scheme
- Given a query, the service provider attaches a *VO* (*Verification Object*) to the results
- The client verifies query results with the *VO* and the owner's signature to ensure:
  - soundness
  - completeness

PRIVACY

- Problem: how to publish data (e.g., for statistical purposes) without disclosing the identity of the records.
- *k*-anonimity
- *Differential privacy*
- Other anonymity concepts
- Anonymity in streams etc

ID	MicroData QI_1	QI_2	SA
A	1	1	v_1
B	1	4	v_2
C	2	2	v_3
D	2	3	v_4
E	3	1	v_5
F	3	2	v_6
G	5	4	v_7

Generalized QI_1	QI_2	SA
1-2	1-4	v_1
1-2	1-4	v_2
1-2	1-4	v_3
1-2	1-4	v_4
3-5	1-4	v_5
3-5	1-4	v_6
3-5	1-4	v_7

QI attributes - quasi-identifiers (e.g., age, address)

SA-sensitive attribute (e.g., dicease, salary)

AI FOR DB

- *Cardinality Estimation using Deep Models*
  - Learned techniques sometimes capture well correlations between attributes (difficult to capture by conventional techniques)
  - Cardinality estimation complicated for joins
- *Database Tuning*
  - Choosing which indexes to build, including learned indexes
  - Query rewriting for optimized performance
  - Select views to materialize
- *Learned DB design*
- *Learned DB security*
- *Learned Transaction Management*

DB FOR AI

- *Data Governance techniques for improving the quality of data in machine learning*
  - Data cleaning, data integration
- *DB based techniques have been used to speed up feature and model selection*
- *DBs can accelerate model execution*

OTHER TOPICS

- Peer-to-Peer and Distributed DB
  - The DB is distributed over several servers
  - How to efficiently store and find the data
  - Minimization of the communication overhead
- Sensor Data Management
  - Related to stream management
  - Minimization of energy consumption
  - In network aggregation, approximation techniques

OTHER TOPICS

- Information Integration
  - How to combine information from several DB
  - Schema matching, Data Cleaning
- Graph Databases
  - How to solve graph problems in huge graphs
  - Clique detection, triangle enumeration, reachability queries etc.
- Social Networks
  - Data managements issues and specialized query processing tasks (influence, clustering based on social connectivity)

OTHER TOPICS

- NOSQL Systems
  - data maintained in means other than tables (key-value stores, column stores, document stores etc.)
- Cloud Databases
  - Database as a service
  - Map-reduce, Hadoop
- Crowdsourcing
  - Use numerous people to perform difficult tasks, such as entity resolution, image matching, and clustering

OTHER TOPICS

- Many more topics - databases touch most aspects of Computer Science
  - Query processing with alternative hardware (e.g., graphic processors).
  - Storage in alternative hardware (e.g., FLASH storage).
  - Algorithms, approximation techniques (e.g., sketches) for large volumes of data.