MauveDB: Statistical Modeling inside Database Systems

Amol Deshpande, University of Maryland

(joint work w/ Sam Madden, MIT)

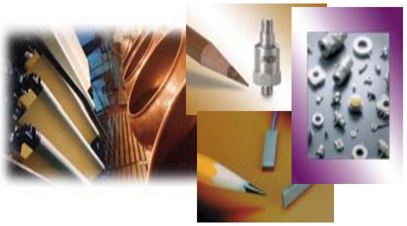
Motivation

- Unprecedented, and rapidly increasing, instrumentation of our every-day world
- Huge data volumes generated *continuously* that must be processed in *real-time*
- Typically *imprecise*, *unreliable* and *incomplete* data
  - Inherent measurement noises (e.g. GPS)
  - Low success rates (e.g. RFID)
  - Communication link or sensor node failures (e.g. wireless sensor networks)
  - Spatial and temporal biases
- Raw sensed data is not what users want to see/query

Wireless sensor networks

Distributed measurement networks (e.g. GPS)

RFID

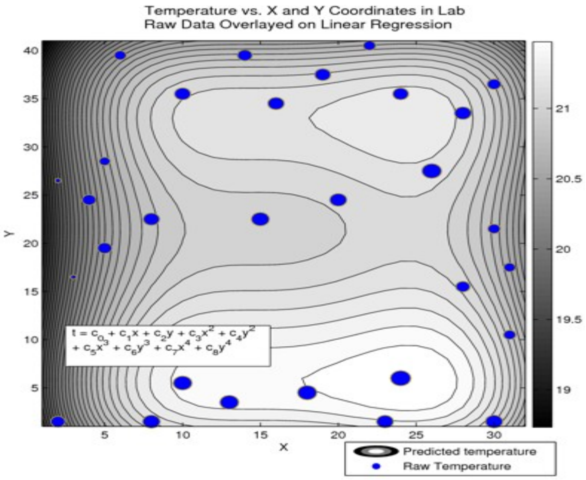


Industrial Monitoring

Data Processing Step 1

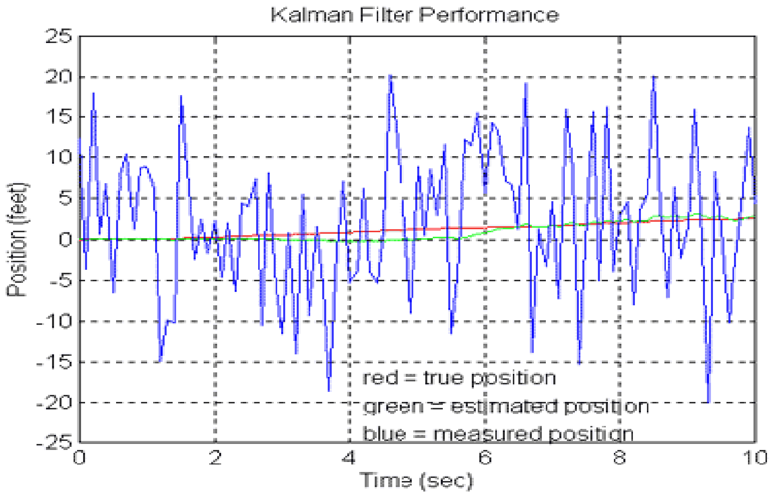
- Process data using a statistical/probabilistic model
  - Regression and interpolation models
    - To eliminate spatial or temporal biases, handle missing data, prediction
  - Filtering techniques *(e.g. Kalman Filters*), Bayesian Networks
    - To eliminate measurement noise, to infer hidden variables etc

Temperature monitoring



Regression/interpolation models

GPS Data



Kalman Filters et

Statistical Modeling of Sensor Data

- No support in database systems --> Database ends up being used as a backing store
  - With much replication of functionality
  - Very inefficient, not declarative…
- How can we push statistical modeling inside a database system ?

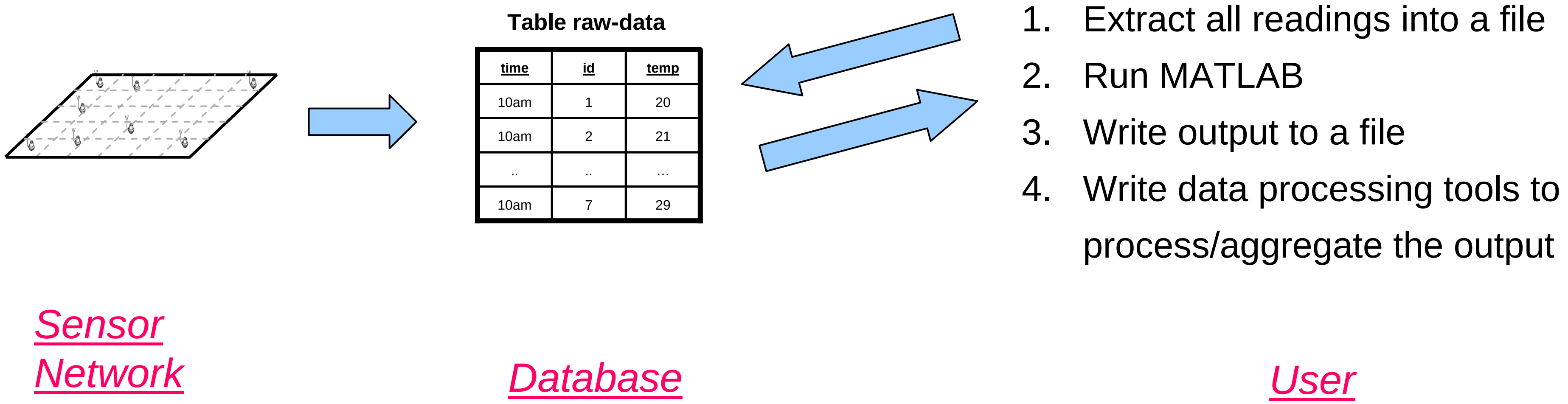


Table raw-data

<u>time</u>	<u>id</u>	<u>temp</u>
10am	1	20
10am	2	21
..	..	…
10am	7	29

1. Extract all readings into a file
2. Run MATLAB
3. Write output to a file
4. Write data processing tools to process/aggregate the output

Sensor Network

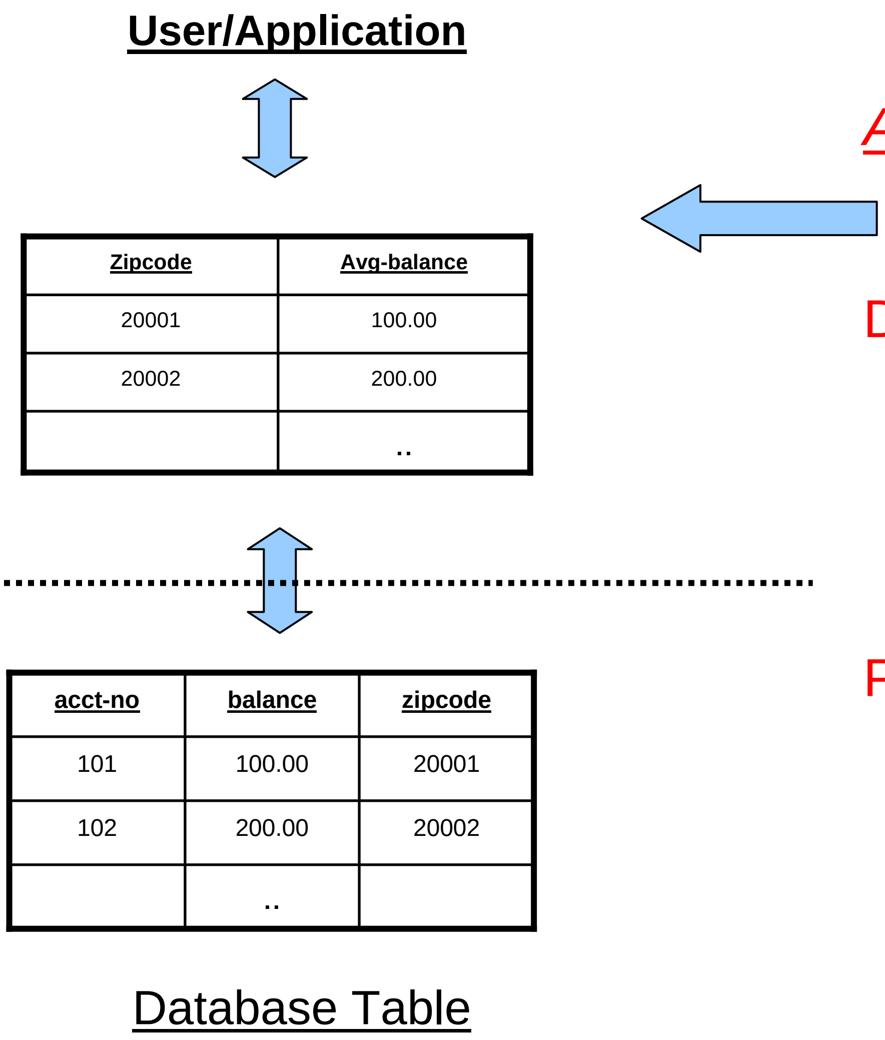
Database

User

Model-based User Views

- An abstraction based on *database views*

Database Views



User/Application

Zipcode	**Avg-balance**
20001	100.00
20002	200.00
	..

acct-no	**balance**	**zipcode**
101	100.00	20001
102	200.00	20002
	..	

Database Table

A Virtual Table

Defined using an SQL Query

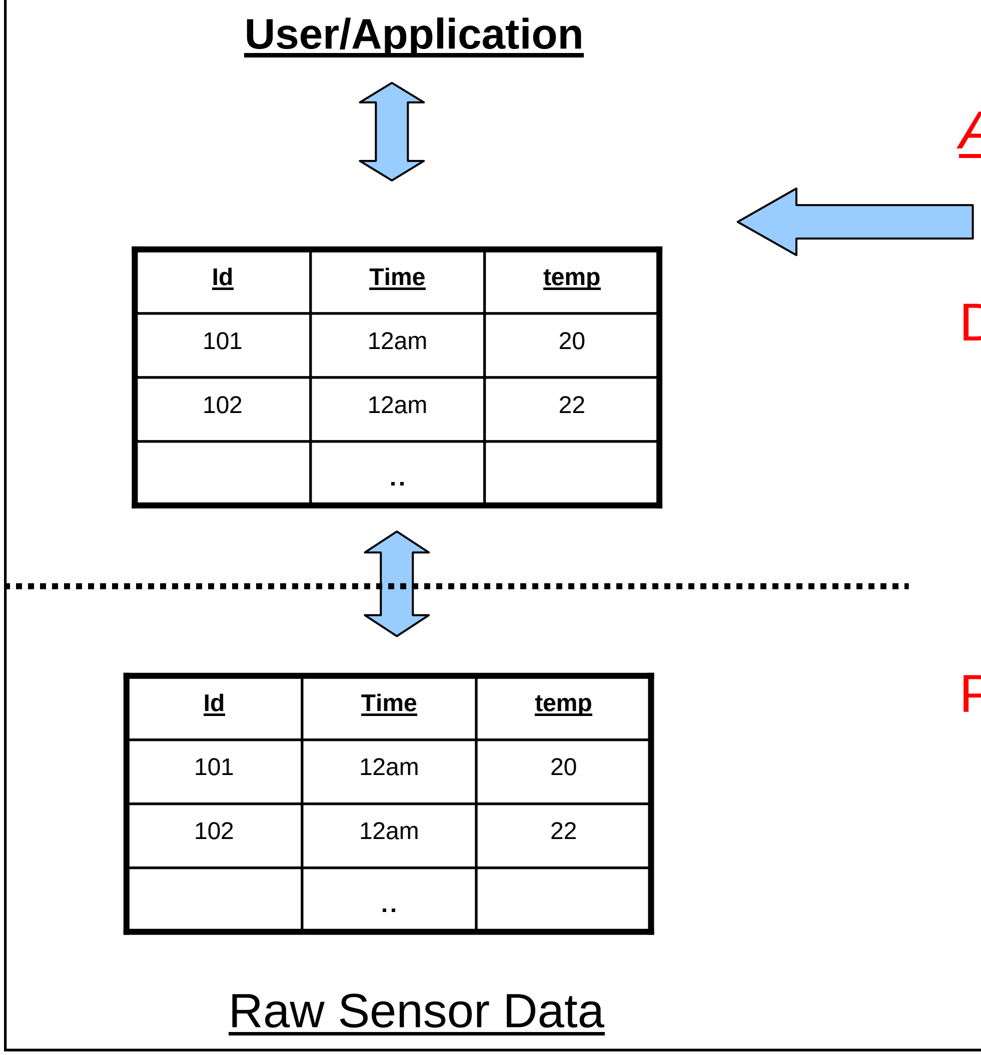
(select zipcode, avg(balance)
from accounts
group by zipcode)

Provides *independence from the details*

(of the schema)

Model-based User Views

- Model-based Views: Define views using statistical models instead



User/Application

Id	Time	temp
101	12am	20
102	12am	22
	..	

Id	Time	temp
101	12am	20
102	12am	22
	..	

Raw Sensor Data

A Virtual Table

Defined using a statistical model

(Use regression to predict missing values, to remove biases, outliers etc)

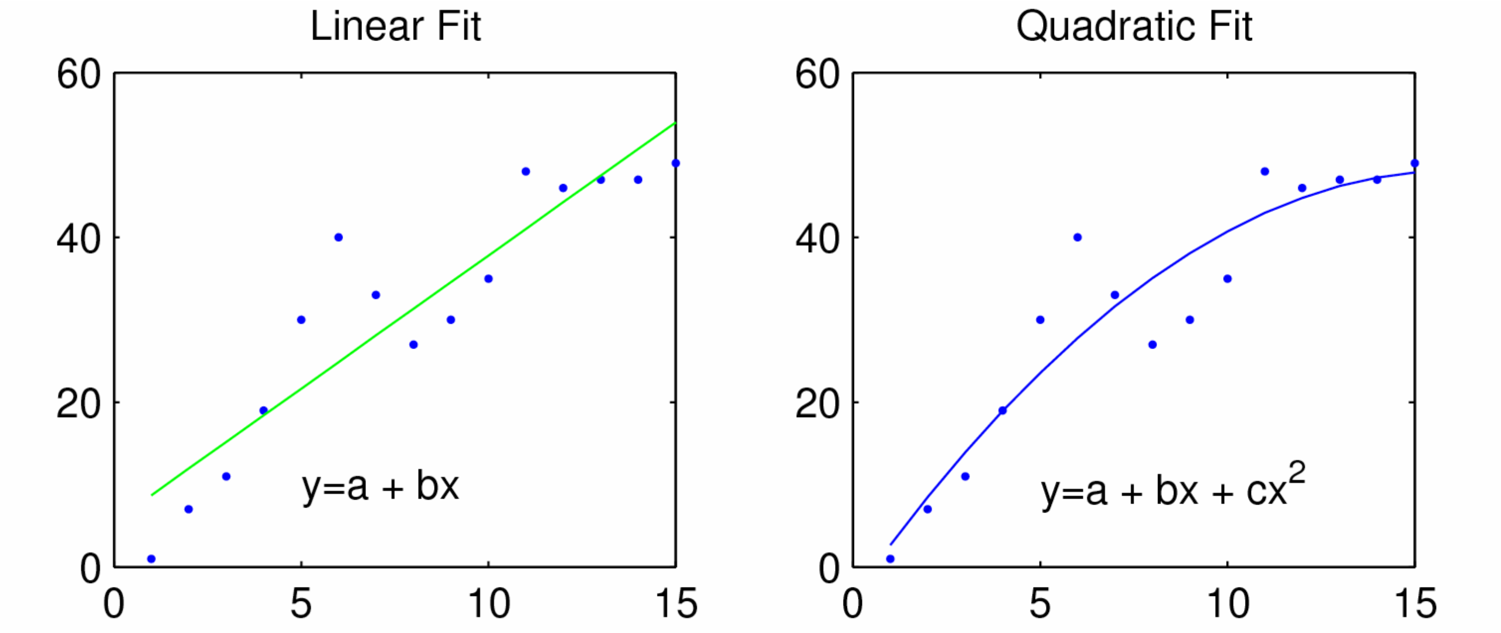
Provides *independence from the details*

(of the measurement infrastructure)

Example: Regression-based Views

Regression:

Model a dependent variable as a function of independent variables

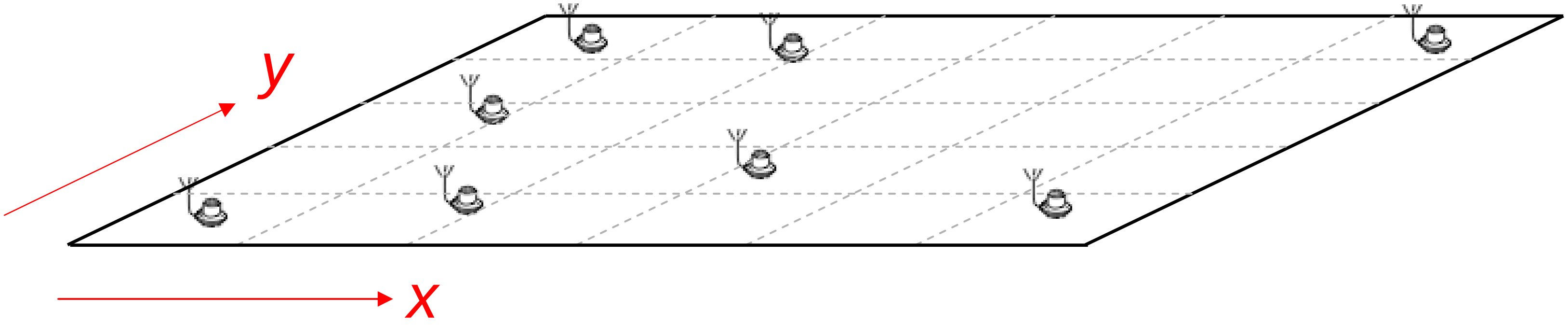


Example: Regression-based Views

Model *temperature* as a function of *(x, y)*

E.g.

$$temp = w_1 + w_2 * x + w_3 * x^2 + w_4 * y + w_5 * y^2$$



Grid Abstraction

QuickTime™ and a
TIFF (LZW) decompressor
are needed to see this picture.

QuickTime™ and a
TIFF (LZW) decompressor
are needed to see this picture.

Creating a Regression-based View

CREATE VIEW

RegView(time [0::1], x [0:100:10], y[0:100:10], temp)

AS

FIT temp USING time, x, y

BASES 1, x, $x^{2,}$ *y,* y^2

FOR EACH time T

TRAINING DATA

SELECT temp, time, x, y

FROM raw-temp-data

WHERE raw-temp-data.time = T

Fit as:

$$temp = w_1 + w_2 * x + w_3 * x^2 + w_4 * y + w_5 * y^2$$

Query Processing

- Analogous to querying database tables
  - *select * from reg-view*
    - Lists out temperatures at all grid-points
  - *select * from reg-view where x = 15 and y = 20*
    - Lists temperature at (15, 20) at all times
  - …
- How are queries evaluated ?
  - Different options
    - Do the statistical modeling it as soon as new data arrives
    - *or* when the queries are asked (on demand)
    - *or* …
  - Optimization opportunities that the database system can exploit
    - Without bothering the user

MauveDB: Status

- Written in the Apache Derby Java open source database system
- Support for *Regression-* and *Interpolation-based views*
  - Currently building support for views based on *Dynamic Bayesian networks (Kalman Filters, HMMs etc)*
- Minimal changes to the main codebase
- Much of the additional code fairly generic in nature
- Model-specific code
  - View creation syntax
  - One of the (four) query processing strategies

Research Challenges/Future Work

- Dynamic *Bayesian* Networks
- Generalizing to arbitrary models ?
  - Develop APIs for adding arbitrary models
  - Try to minimize the work of the model developer
- *Probabilistic databases*
  - Uncertain data with complex correlation patterns
- Query processing, query optimization
- View maintenance in presence of high-rate measurement streams