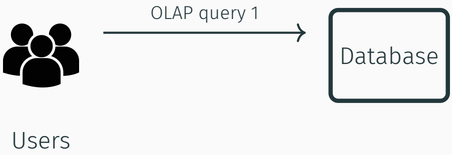
DATABASE LEARNING:

Toward a Database that Becomes Smarter Every Time

Yongjoo Park
Ahmad Shahab Tajik
Michael Cafarella
Barzan Mozafari

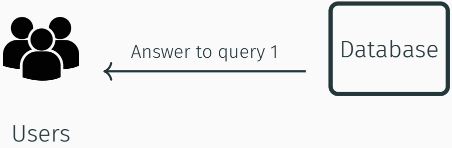
University of Michigan, Ann Arbor

Today’s Databases

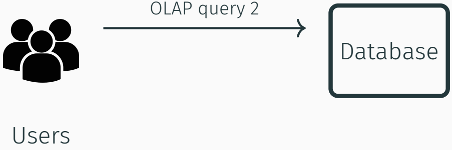


Today’s Databases

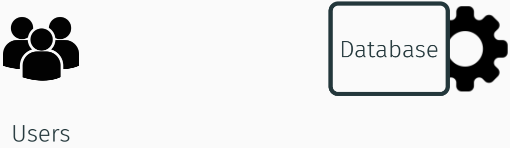
Today's Databases



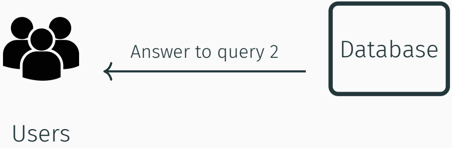
Today's Databases



Today's Databases



Today's Databases

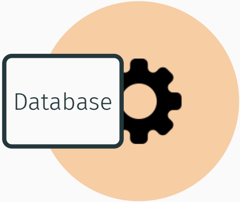


Today's Databases

After answering queries,
THE WORK is almost completely WASTED.

Today's Databases

Users



After answering queries,

THE WORK is almost completely WASTED.

Small exceptions:

Today’s Databases

After answering queries,
THE WORK is almost completely **WASTED**.

Small exceptions:

- Caching

Today's Databases

After answering queries,
THE WORK is almost completely WASTED.

Small exceptions:

- Caching
- Identical queries

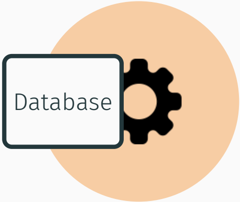
Today's Databases

After answering queries,
THE WORK is almost completely WASTED.

Small exceptions:

- Caching
- Identical queries
- Indexing/Materialization hints

Today's Databases



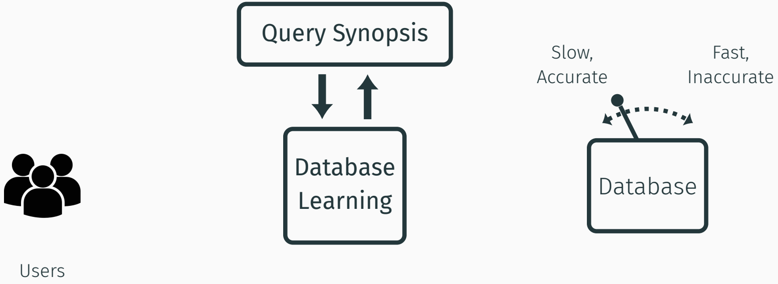
After answering queries,
THE WORK is almost completely **WASTED**.

Small exceptions:

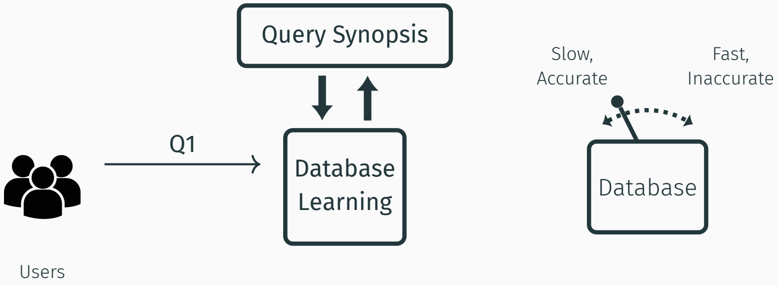
- Caching
- Identical queries
- Indexing/Materialization hints

Our Goal: reuse **the work.**

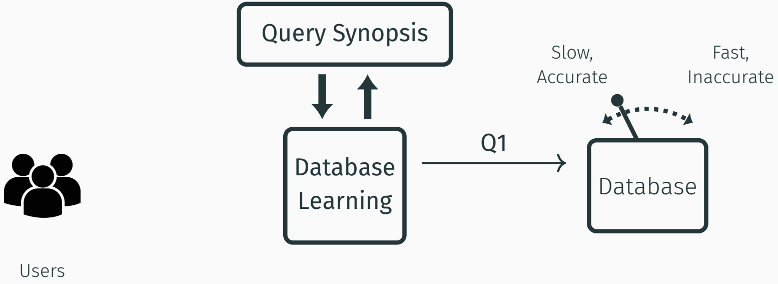
A New Paradigm



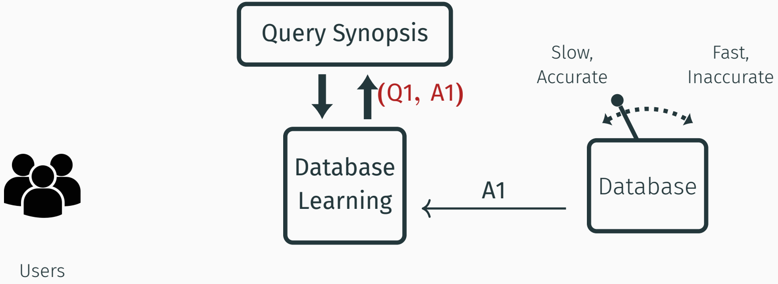
A New Paradigm



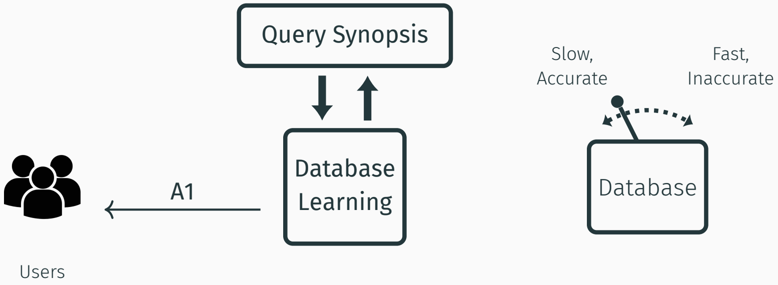
A New Paradigm



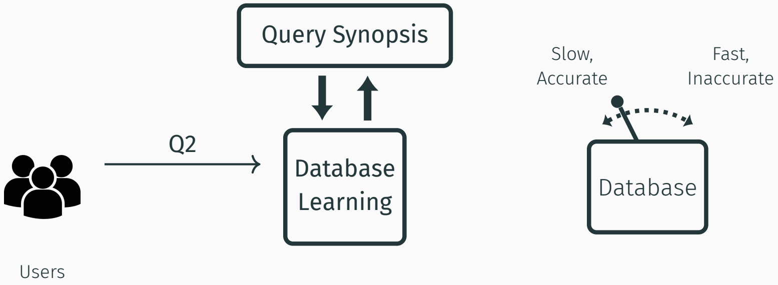
A New Paradigm



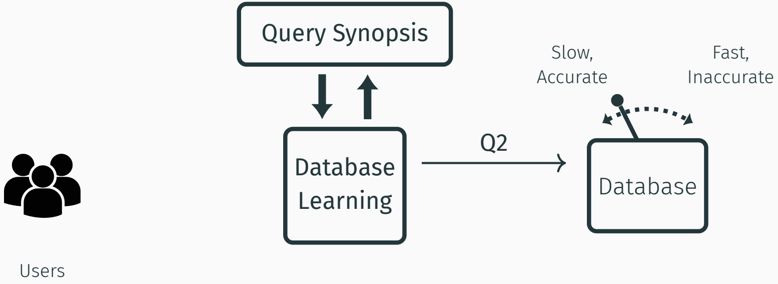
A New Paradigm



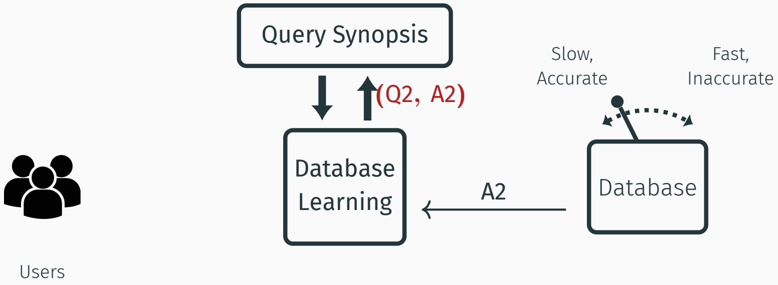
A New Paradigm



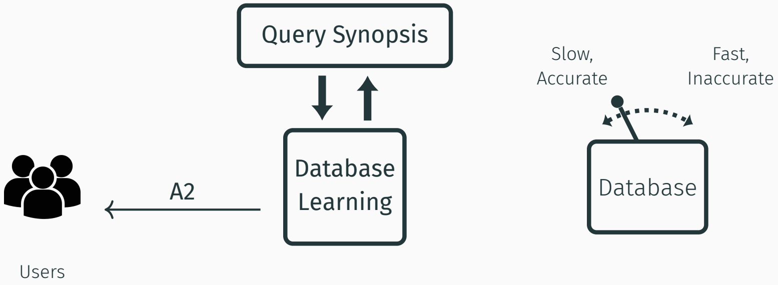
A New Paradigm



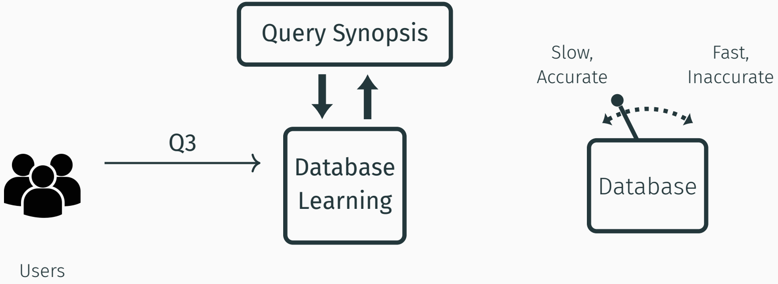
A New Paradigm



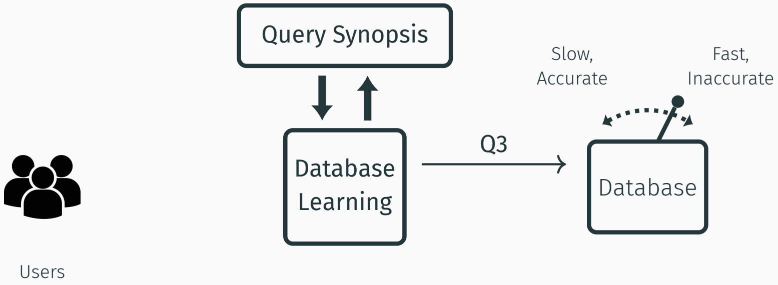
A New Paradigm



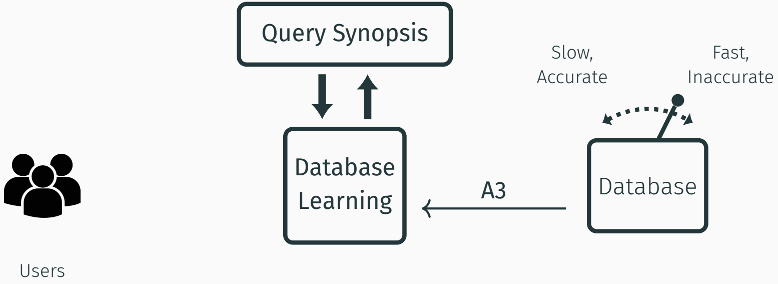
A New Paradigm



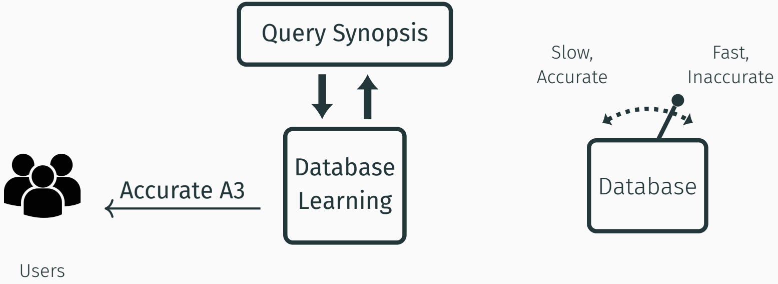
A New Paradigm



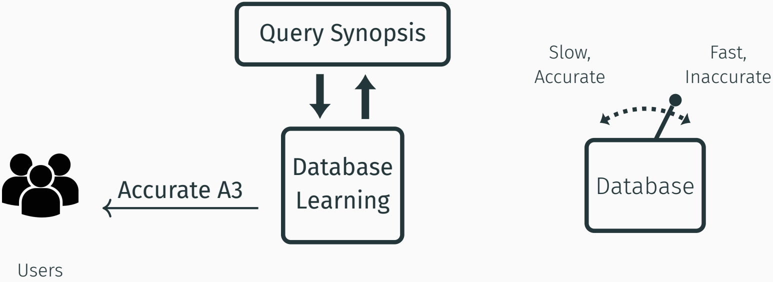
A New Paradigm



A New Paradigm

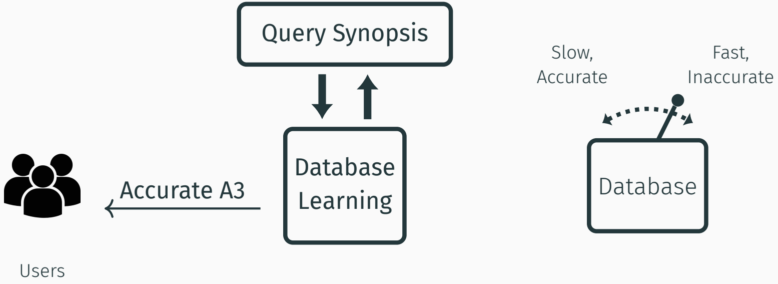


A New Paradigm



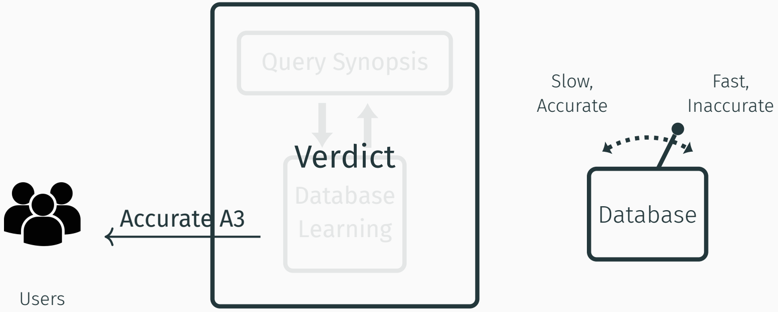
1. **Never lower** speed or accuracy

A New Paradigm



1. Never lower speed or accuracy
2. Popularity of analytic workloads ⇒ **Approximate solutions**
    - BlinkDB, SnappyData, Yahoo Druid, Facebook Presto, etc.

A New Paradigm



1. **Never lower** speed or accuracy
2. Popularity of analytic workloads ⇒ **Approximate solutions**
    - BlinkDB, SnappyData, Yahoo Druid, Facebook Presto, etc.
3. [Verdict, CIDR'15]: DB Learning with **any RDBMS**
    - Vertica, SparkSQL, Oracle, TeraData, and so on.

From Machine Learning To Database Learning

Machine Learning: Past Observations $\Rightarrow$ Future Predictions

From Machine Learning To Database Learning

Machine Learning:	Past Observations	$\Rightarrow$	Future Predictions
Database Learning:	**Past Answers**	$\Rightarrow$	**Future Answers**

From Machine Learning To Database Learning

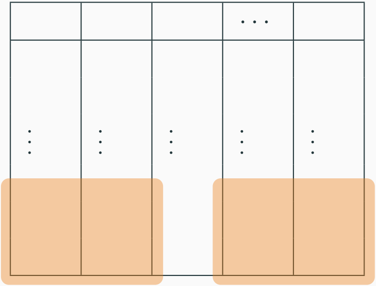
Machine Learning:	Past Observations	$\Rightarrow$	Future Predictions
Database Learning:	**Past Answers**	$\Rightarrow$	**Future Answers**

The more past queries, the more **Accurate** and **Faster**

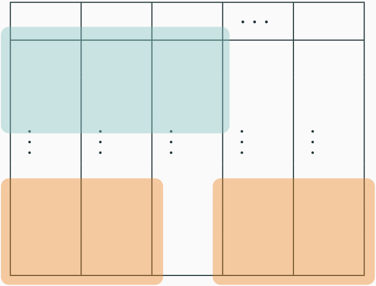
Exploratory Workloads

			…	
⋮	⋮	⋮	⋮	⋮

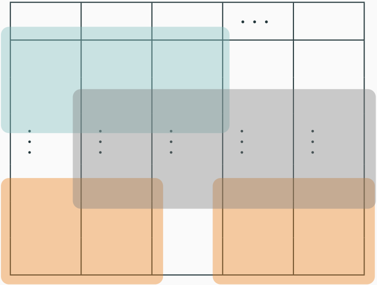
Exploratory Workloads



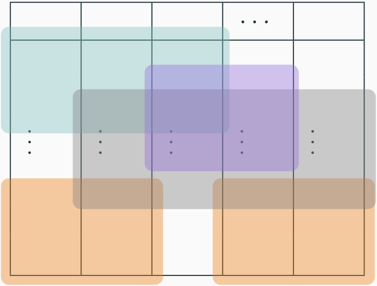
Exploratory Workloads



Exploratory Workloads

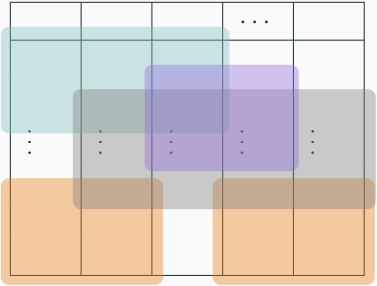


Exploratory Workloads



Queries use the data in different columns/rows.

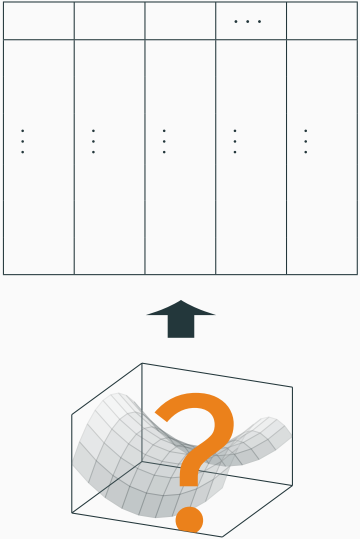
Exploratory Workloads



Queries use the data in different columns/rows.

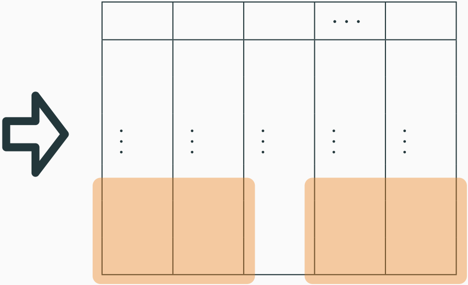
How can those queries help each other?

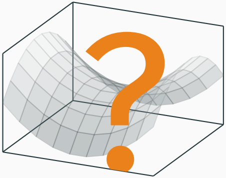
Our Idea



Our Idea

Q1

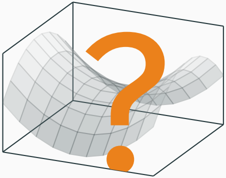




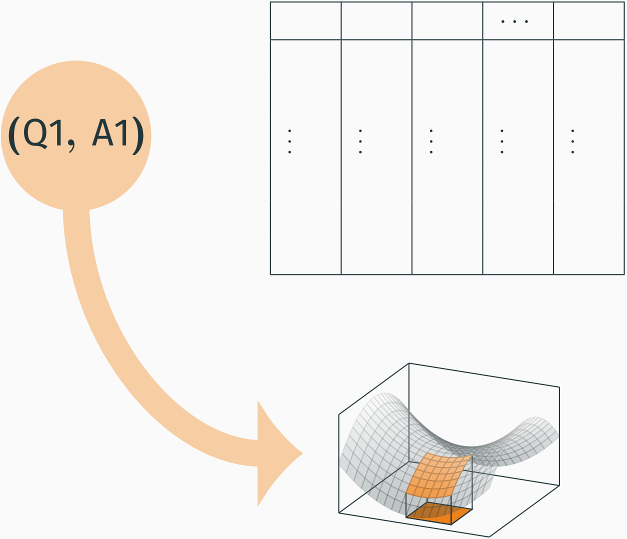
Our Idea

(Q1, A1)

			$\cdots$	
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

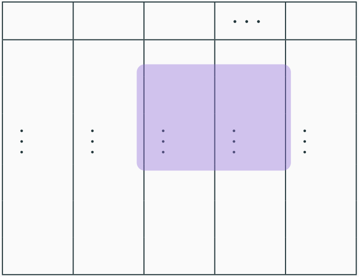


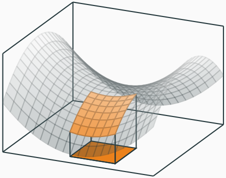
Our Idea



Our Idea

Q2

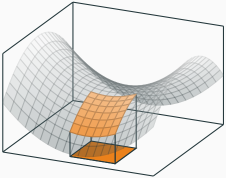




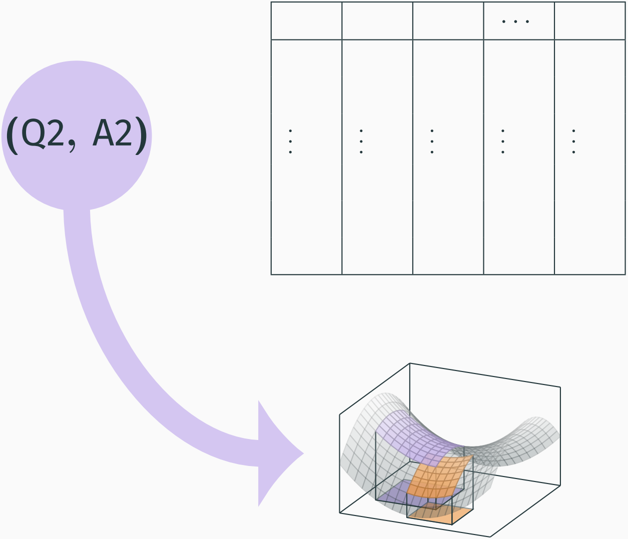
Our Idea

(Q2, A2)

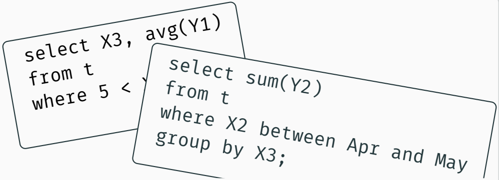
			…	
⋮	⋮	⋮	⋮	⋮



Our Idea

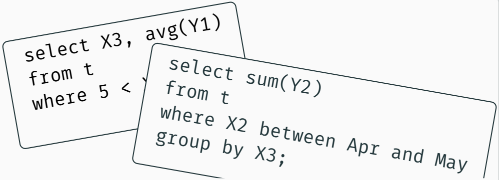


Design Criteria of Database Learning

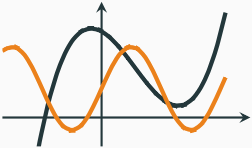


1. Support a **wide class of** SQL queries

Design Criteria of Database Learning

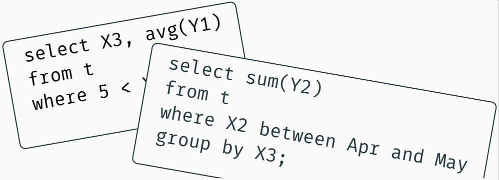


1. Support a **wide class of** SQL queries

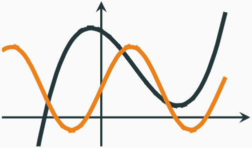


2. **No Assumptions** about Data

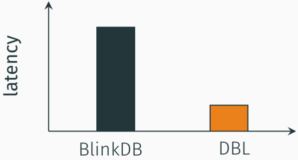
Design Criteria of Database Learning



1. Support a **wide class of** SQL queries



2. **No Assumptions** about Data



3. Lightweight

How to achieve Database Learning

```
select avg(Y2)
from t
where 6 < X1 < 8;
```

How to achieve Database Learning

```
select sum(Y2)
from t
where 5 < X1 < 8;
```

How to achieve Database Learning

```
select sum(Y2)
from t
where 5 < X1 < 8;
```

1

Random variables

$\theta_1, \quad \theta_2, \quad \theta_3$

How to achieve Database Learning

```
select sum(Y2)
from t
where 5 < X1 < 8;
```

1

Random variables

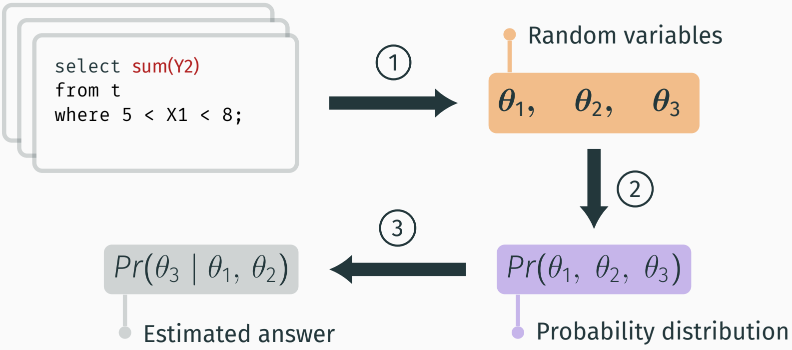
$\theta_1, \quad \theta_2, \quad \theta_3$

2

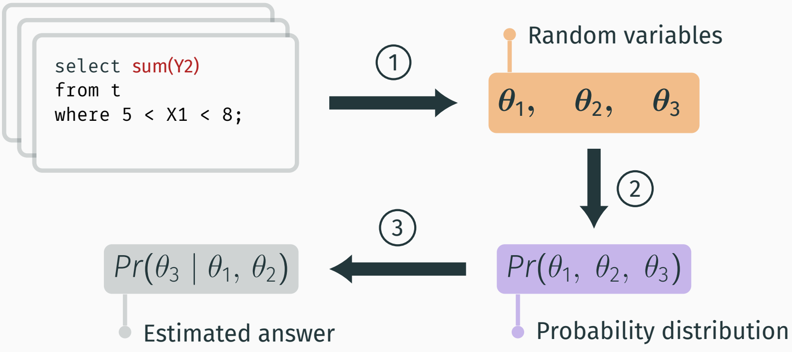
$Pr(\theta_1,\ \theta_2,\ \theta_3)$

Probability distribution

How to achieve Database Learning

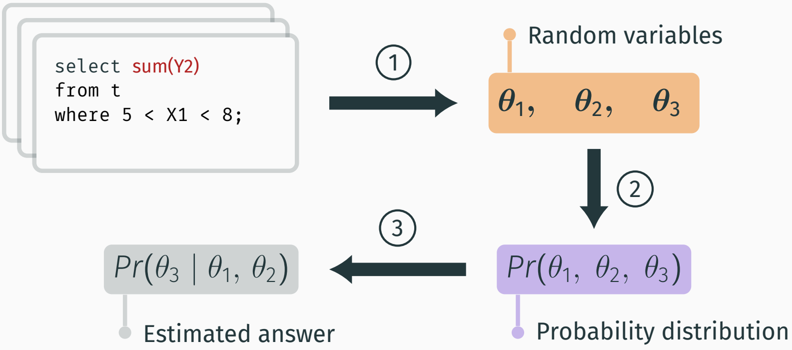


How to achieve Database Learning



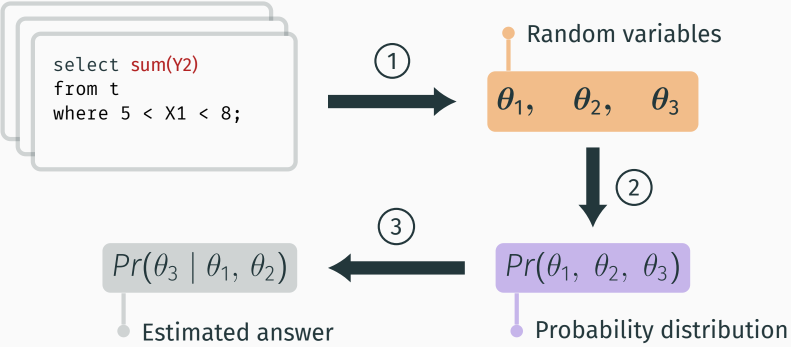
1. No assumptions about data

How to achieve Database Learning



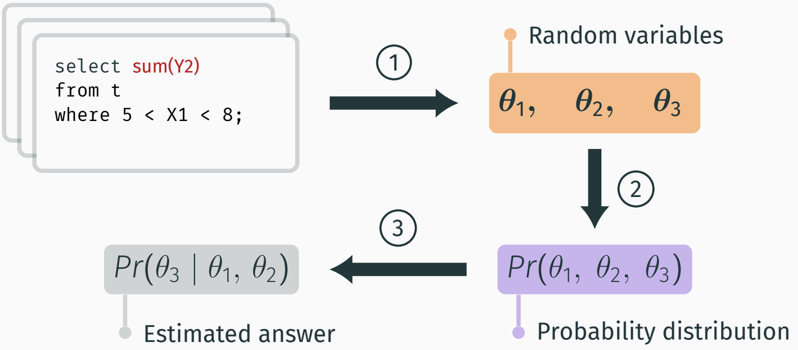
1. **No assumptions** about data
2. Important questions:

How to achieve Database Learning



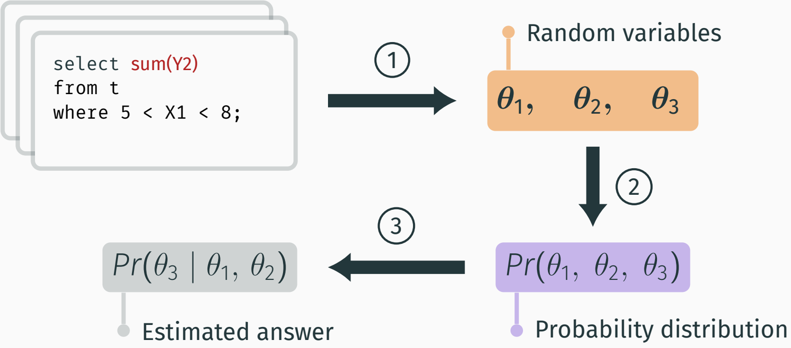
1. No assumptions about data
2. Important questions:
    - How to define random variables

How to achieve Database Learning



1. **No assumptions** about data
2. Important questions:
    - How to define random variables
    - How to determine probability distribution

How to achieve Database Learning



1. No assumptions about data
2. Important questions:
    - How to define random variables
    - How to determine probability distribution
    - How to make inference fast

First Question: How to define Random Variable

```
select sum(Y2)
from t
where 5 < X1 < 8;
```

First Question: How to define Random Variable

```
select sum(Y2)
from t
where 5 < X1 < 8;
```

We define a random variable θ
for every combination of:

First Question: How to define Random Variable

```
select sum(Y2)
from t
where 5 < X1 < 8;
```

We define a random variable θ for every combination of:

- Aggregate function

First Question: How to define Random Variable

We define a random variable θ for every combination of:

```
select sum(Y2)
from t
where 5 < X1 < 8;
```

- Aggregate function
- Selection predicates

First Question: How to define Random Variable

```
select sum(Y2)
from t
where 5 < X1 < 8;
```

We define a random variable θ for every combination of:

- Aggregate function
- Selection predicates

```
select X3, avg(Y1), sum(Y2)
from t
where 5 < X1 < 8
  and X2 between Apr and May
group by X3;
```

What if your query is complex?

First Question: How to define Random Variable

```
select sum(Y2)
from t
where 5 < X1 < 8;
```

We define a random variable θ for every combination of:

- Aggregate function
- Selection predicates

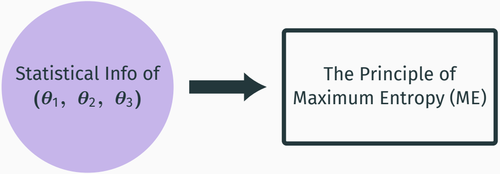
```
select X3, avg(Y1), sum(Y2)
from t
where 5 < X1 < 8
  and X2 between Apr and May
group by X3;
```

What if your query is complex?

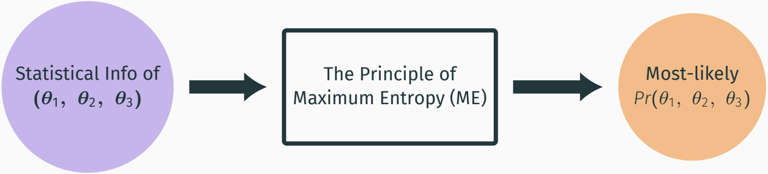
Second Question: How to determine Probability Distribution

The Principle of Maximum Entropy (ME)

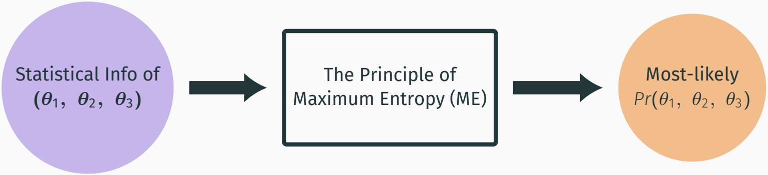
Second Question: How to determine Probability Distribution



Second Question: How to determine Probability Distribution

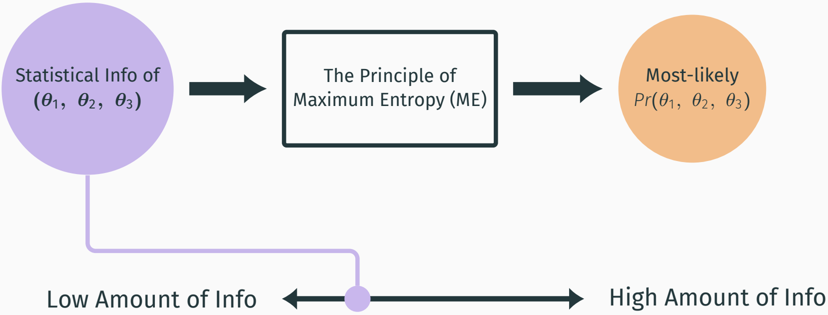


Second Question: How to determine Probability Distribution

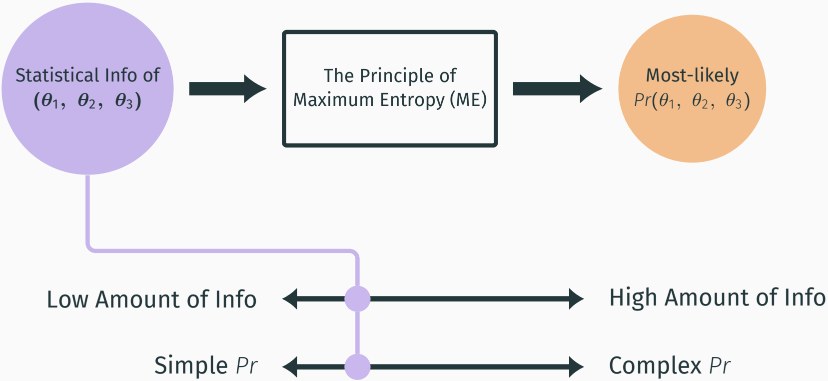


Low Amount of Info $\longleftrightarrow$ High Amount of Info

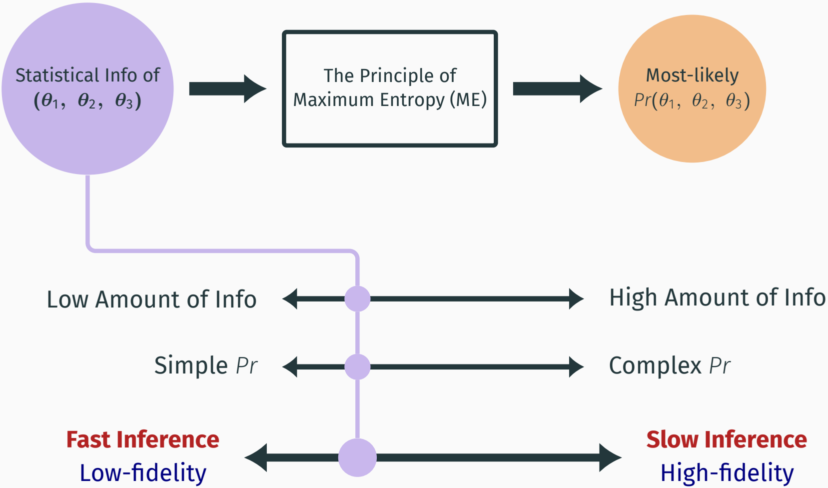
Second Question: How to determine Probability Distribution



Second Question: How to determine Probability Distribution



Second Question: How to determine Probability Distribution



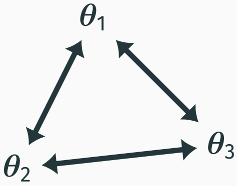
Third Question: How to make Inference Fast

θ_1

θ_3

θ_2

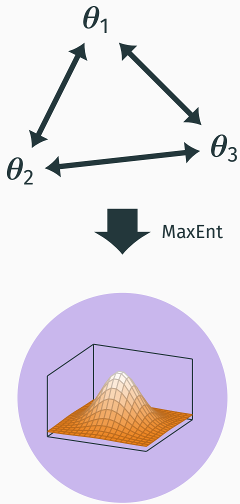
Third Question: How to make Inference Fast



Statistical Information:

Mean, Variances, Covariances

Third Question: How to make Inference Fast

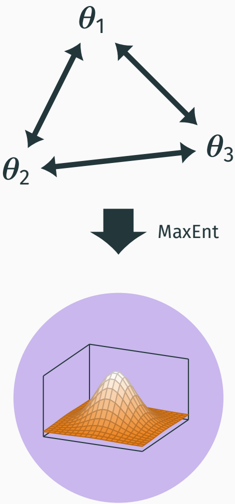


Statistical Information:

Mean, Variances, Covariances

Multivariate Normal Distribution

Third Question: How to make Inference Fast



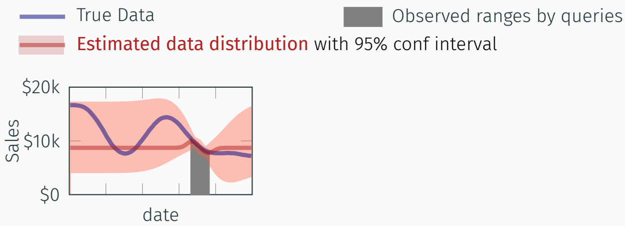
Statistical Information:

Mean, Variances, Covariances

Multivariate Normal Distribution

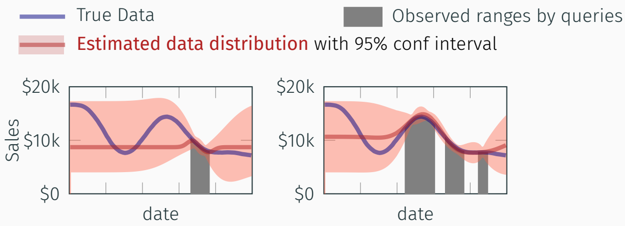
Fast inference using a closed form

Example of Database Learning



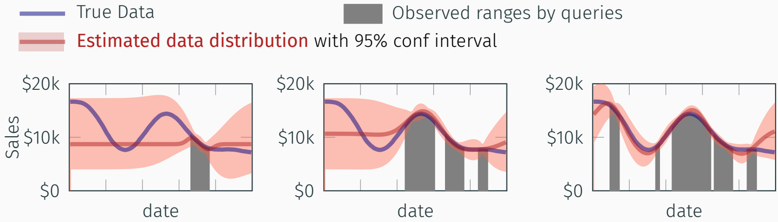
1. Knowledge is probabilistic → provides confidence interval.

Example of Database Learning



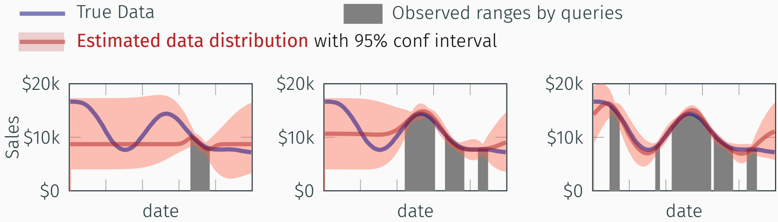
1. Knowledge is probabilistic $\rightarrow$ provides confidence interval.
2. Improves as processing more queries.

Example of Database Learning



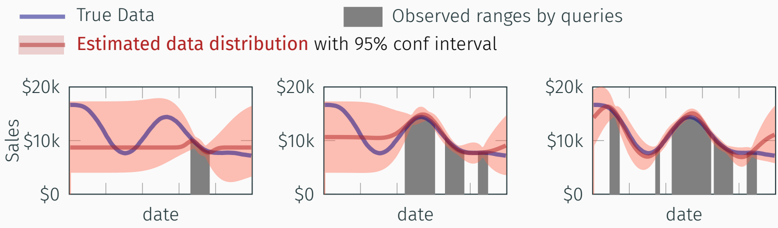
1. Knowledge is probabilistic → provides confidence interval.
2. Improves as processing more queries.

Example of Database Learning



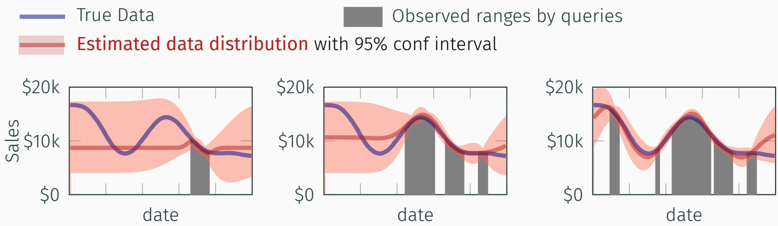
1. **Knowledge** is probabilistic → provides confidence interval.
2. Improves **as processing more queries**.
3. Generalizes to

Example of Database Learning



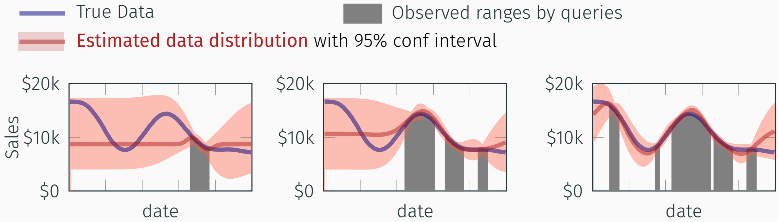
1. Knowledge is probabilistic → provides confidence interval.
2. Improves as processing more queries.
3. Generalizes to
   - tables of large dimensions

Example of Database Learning



1. Knowledge is probabilistic → provides confidence interval.
2. Improves as processing more queries.
3. Generalizes to
   - tables of large dimensions
   - various selection predicates

Example of Database Learning



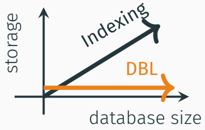
1. **Knowledge** is probabilistic → provides confidence interval.
2. Improves **as processing more queries**.
3. Generalizes to
   - tables of large dimensions
   - various selection predicates
   - different aggregate functions

Benefits of Database Learning

Database Learning vs. Indexing

Benefits of Database Learning

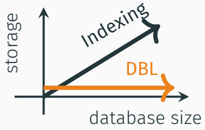
Database Learning vs. Indexing



1. Little Storage Overhead

Benefits of Database Learning

Database Learning vs. Indexing

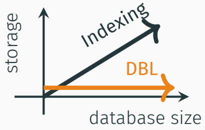


1. Little Storage Overhead

Database Learning vs. View Selection

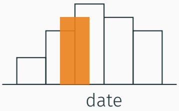
Benefits of Database Learning

Database Learning vs. Indexing



1. Little Storage Overhead

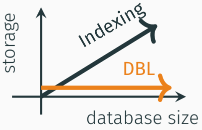
Database Learning vs. View Selection



2. Without Alignment

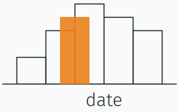
Benefits of Database Learning

Database Learning vs. Indexing

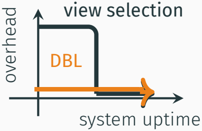


1. Little Storage Overhead

Database Learning vs. View Selection



2. Without Alignment



3. No Upfront Overhead

1. Using **Spark SQL** as a backend

Implementation and Experiments

1. Using **Spark SQL** as a backend
2. Three systems:

Implementation and Experiments

1. Using **Spark SQL** as a backend
2. Three systems:
   - No Sampling **(SparkSQL)**

Implementation and Experiments

1. Using **Spark SQL** as a backend
2. Three systems:
   - No Sampling **(SparkSQL)**
   - BlinkDB on SparkSQL **(BlinkOnSpark)**

Implementation and Experiments

1. Using **Spark SQL** as a backend
2. Three systems:
   - No Sampling **(SparkSQL)**
   - BlinkDB on SparkSQL **(BlinkOnSpark)**
   - Database Learning on BlinkOnSpark **(DBL)**

Implementation and Experiments

1. Using **Spark SQL** as a backend
2. Three systems:
   - No Sampling **(SparkSQL)**
   - BlinkDB on SparkSQL **(BlinkOnSpark)**
   - Database Learning on BlinkOnSpark **(DBL)**

1. 100G TPC-H dataset

Implementation and Experiments

1. Using **Spark SQL** as a backend
2. Three systems:
   - No Sampling **(SparkSQL)**
   - BlinkDB on SparkSQL **(BlinkOnSpark)**
   - Database Learning on BlinkOnSpark **(DBL)**

1. 100G TPC-H dataset
2. 20 Amazon EC2 workers

Speed Improvement