Learning-based Cost Management for Cloud Databases

Olga Papaemmanouil
Brandeis University

Outline

- Motivation
- Offline Learning
- Online Learning
- Conclusions

Outline

- Motivation
- Offline Learning
- Online Learning
- Conclusions

❑ Cloud Databases

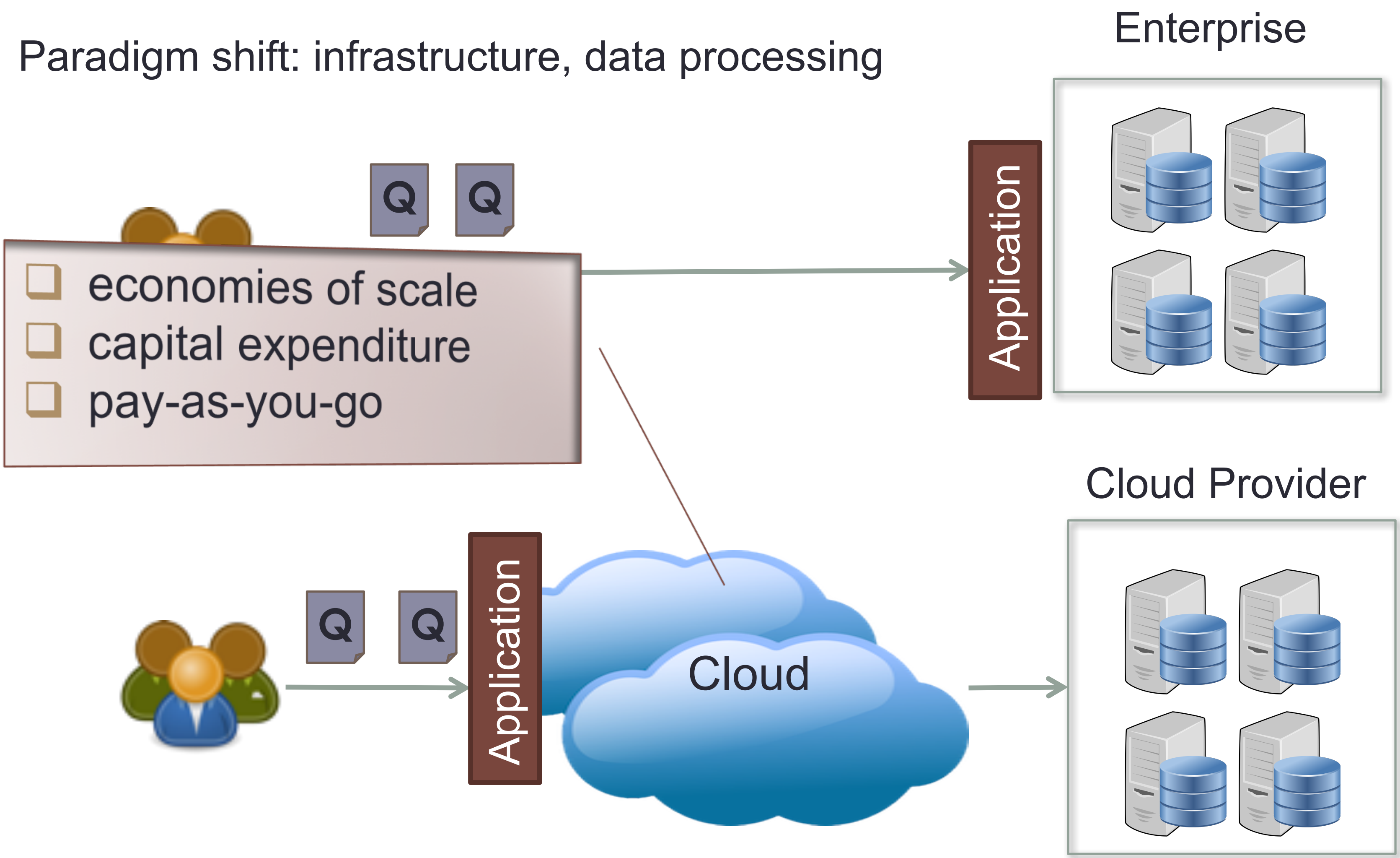
❑ Challenges

❑ Why Machine Learning ?

Cloud Computing

Paradigm shift: infrastructure, data processing

- economies of scale
- capital expenditure
- pay-as-you-go



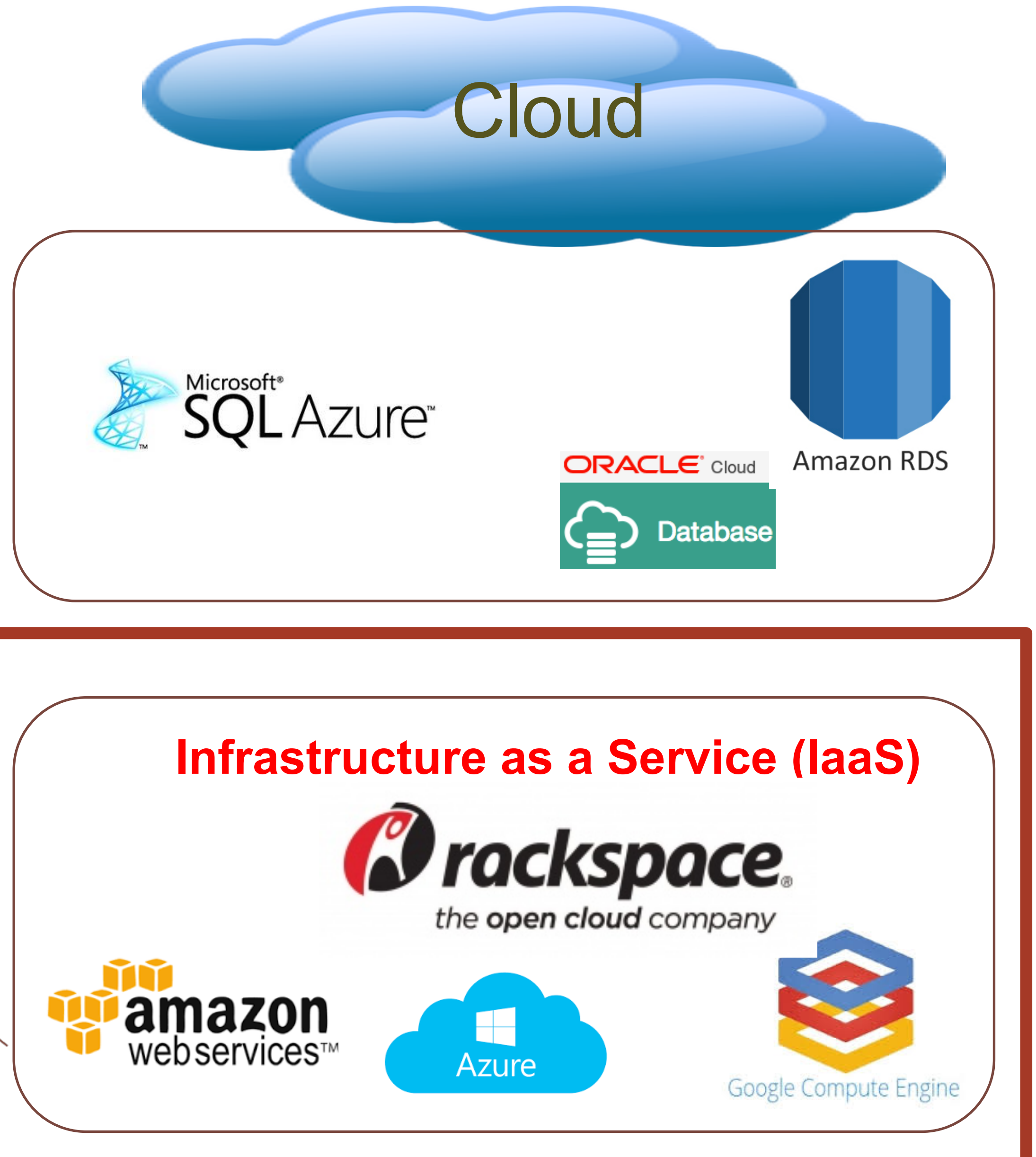
Cloud Databases Landscape

Database-as-a-Service

- Managed DBMS
- Relational & NoSQL DBs

IaaS-based DB Instances

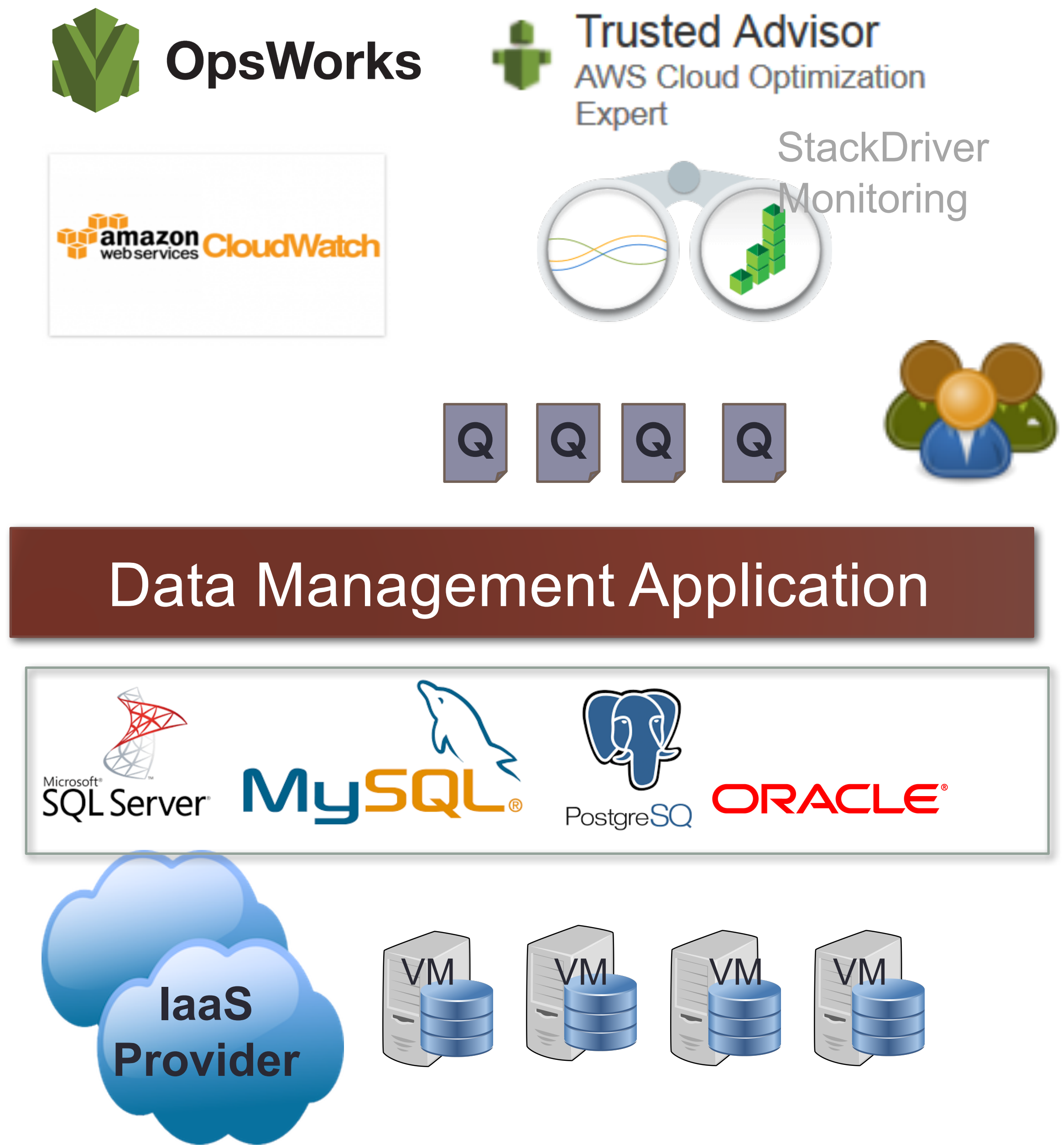
- Non managed DBMS
- Do It Yourself model



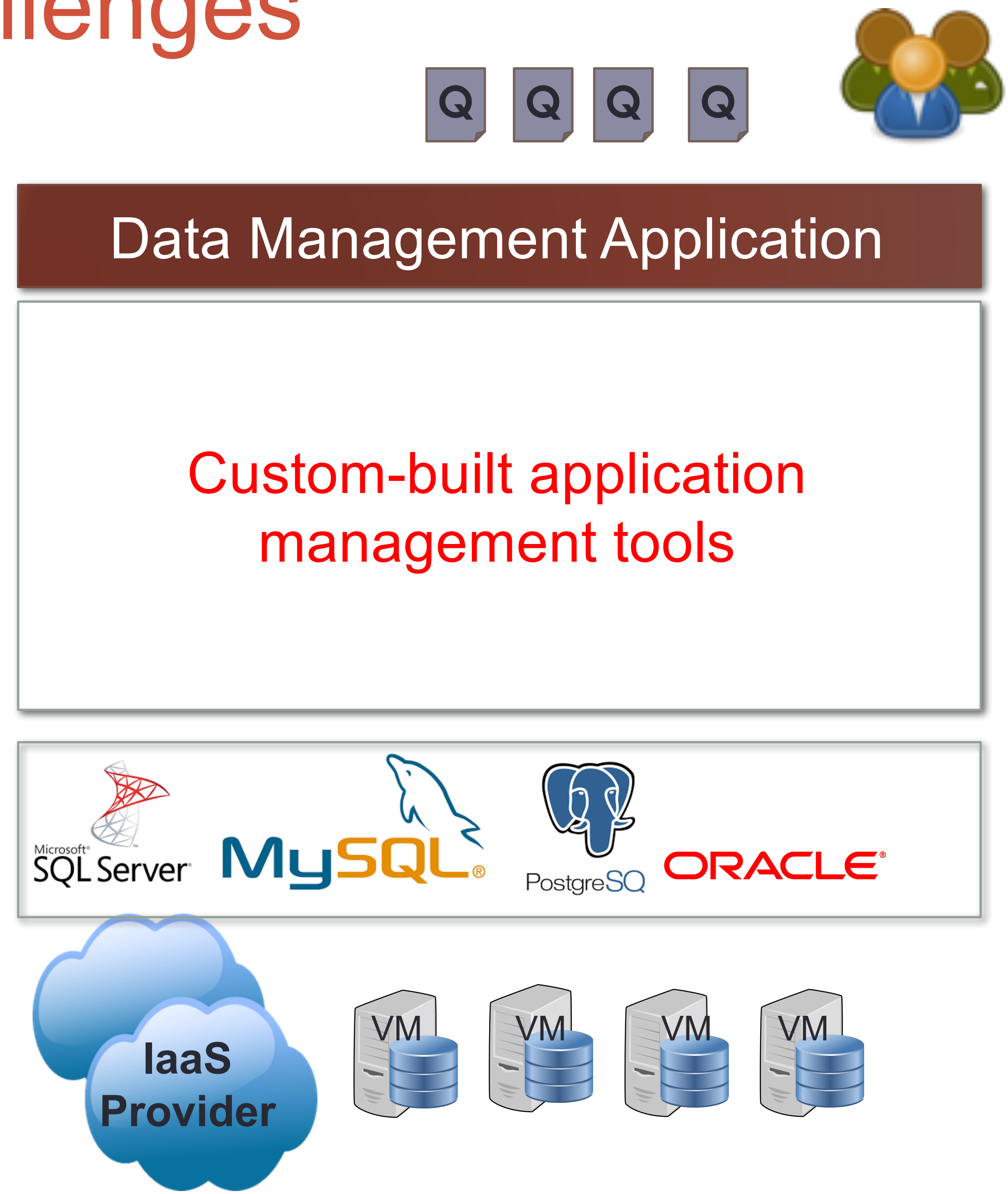
IaaS-deployed Databases

App Management Tools

- Monitoring resources, performance, cost
- Event-driven scaling
- NO cost vs performance optimization



Deployment Challenges



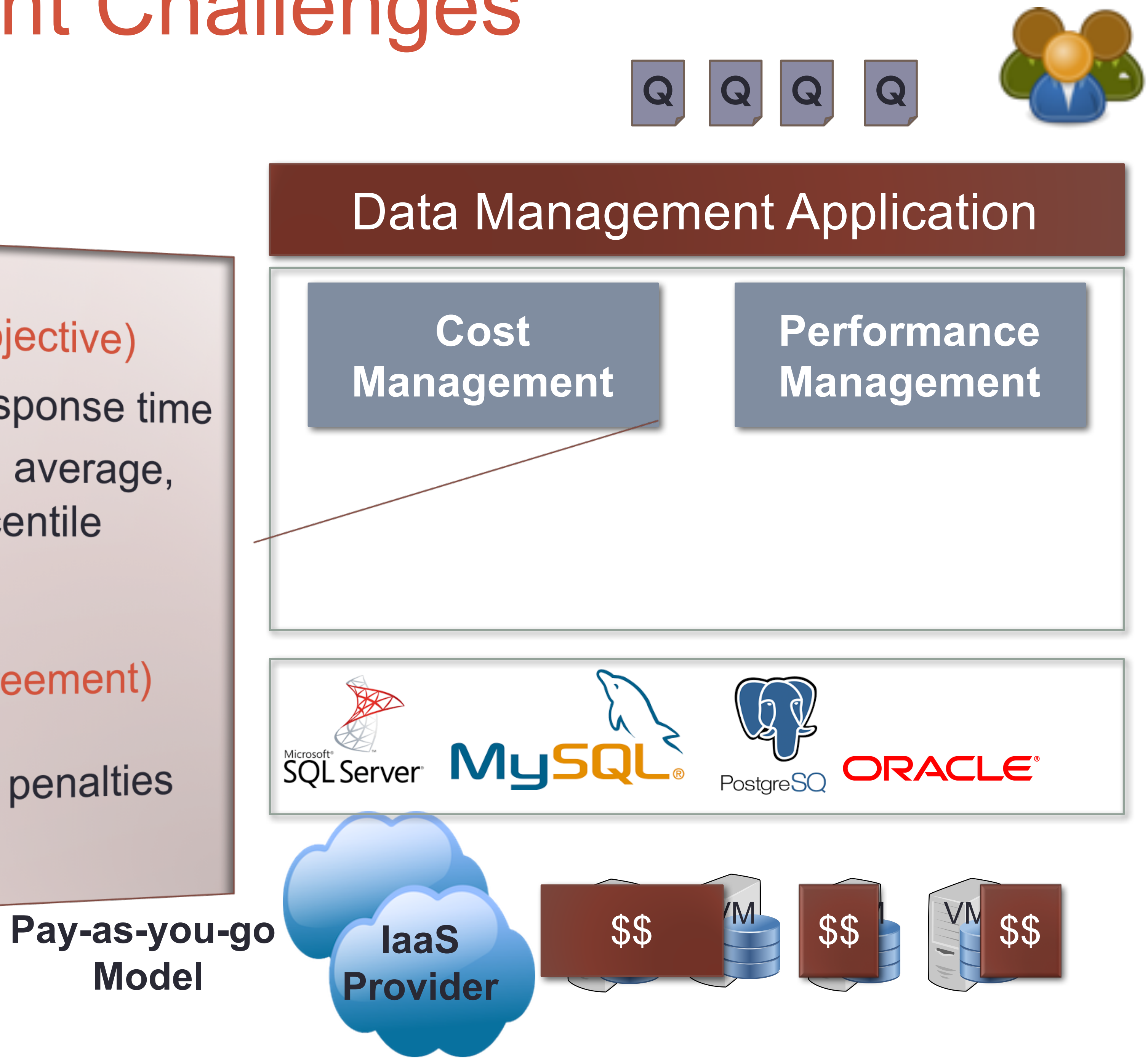
Deployment Challenges

Meet SLOs
(Service Level Objective)

- ❑ Query-level: response time
- ❑ Workload level: average, total, max, percentile

Offer SLAs
(Service Level Agreement)

- ❑ SLO+ Violation penalties

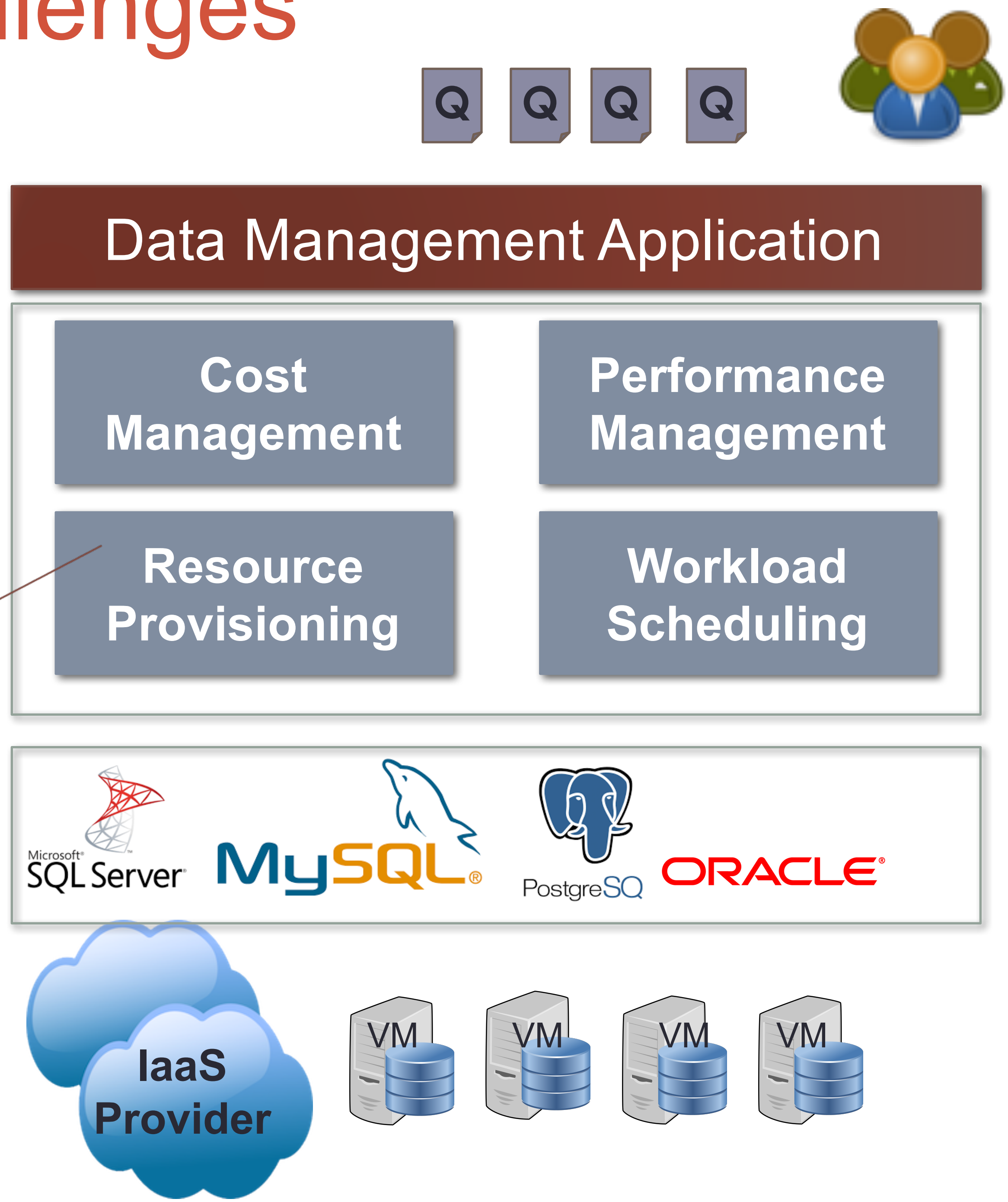


Deployment Challenges

NP-hard problem

Beyond monitoring & alerts

- Automatic scale up & down
- Query routing & scheduling
- Cost-driven decisions
- SLA-awareness



State-of-the-art

<table>
<tr><th>Placement</th><th colspan="2">Provisioning</th><th>Scheduling</th></tr>
<tr><td>PMAX
(Liu et al.)</td><td>Auto
(Rogers et al.)</td><td>Dolly
(Cecchet et all)</td><td>Shepherd
(Chi et al.)</td></tr>
<tr><td colspan="4">SLATree
(Chi et al.)</td></tr>
<tr><td colspan="3">Multi-tenant SLOs
(Lang et al.)</td><td>iCBS
(Chi et al.)</td></tr>
<tr><td>Delphi / Pythia
(Elmore et al.)</td><td colspan="3">Hypergraph
(Çatalyürek et al.)</td></tr>
<tr><td>SCOPE
(Chaiken et al.)</td><td colspan="2">Bazaar
(Jalaparti et al.)</td><td>many traditional methods ...</td></tr>
</table>

State-of-the-art

- Query deadline
- Average latency
- Workload deadline
- Percentile deadline
- Piecewise linear

<table>
<tr><th>Placement</th><th colspan="2">Provisioning</th><th>Scheduling</th></tr>
<tr><td>PMAX
(Liu et al.)</td><td>Auto
(Rogers et al.)</td><td>Dolly
(Cecchet et all)</td><td>Shepherd
(Chi et al.)</td></tr>
<tr><td colspan="4">SLATree
(Chi et al.)</td></tr>
<tr><td colspan="3">Multi-tenant SLOs
(Lang et al.)</td><td>iCBS
(Chi et al.)</td></tr>
<tr><td>Delphi / Pythia
(Elmore et al.)</td><td colspan="3">Hypergraph
(Çatalyürek et al.)</td></tr>
<tr><td>SCOPE
(Chaiken et al.)</td><td colspan="2">Bazaar
(Jalaparti et al.)</td><td>many traditional methods ...</td></tr>
</table>

Wish List

Challenges

End-to-end cost-aware service (resource provisioning, workload scheduling)	**complex interactions**
Application-defined performance goals (per query deadline, percentile, average latency, max latency)	**arbitrary goals**
Agnostic to workload semantics	**arbitrary workloads**

machine learning: auto modeling and insight

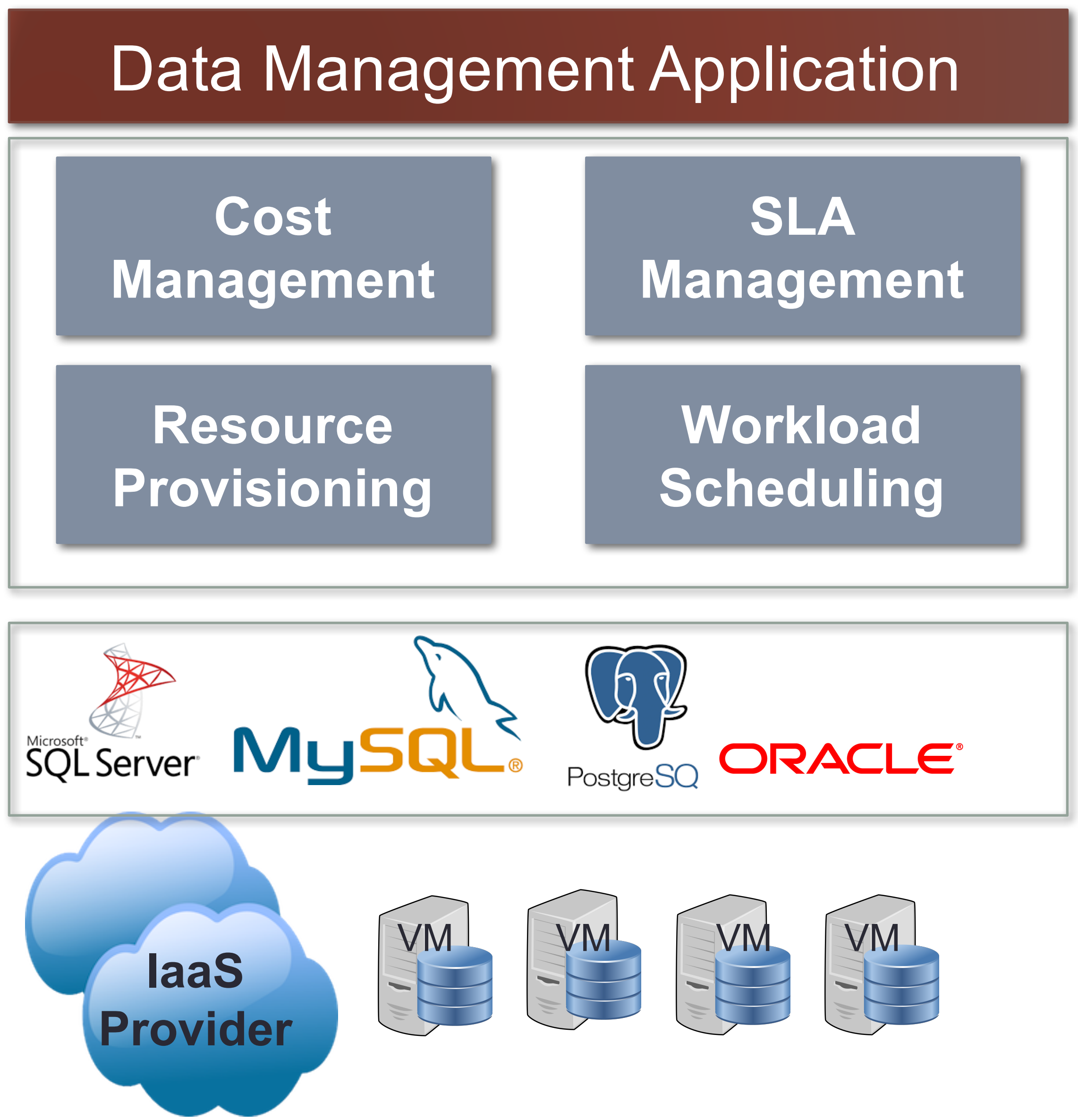
WiSeDB Advisor

Offline Learning

- batch scheduling

Online Learning

- online scheduling
- performance model free

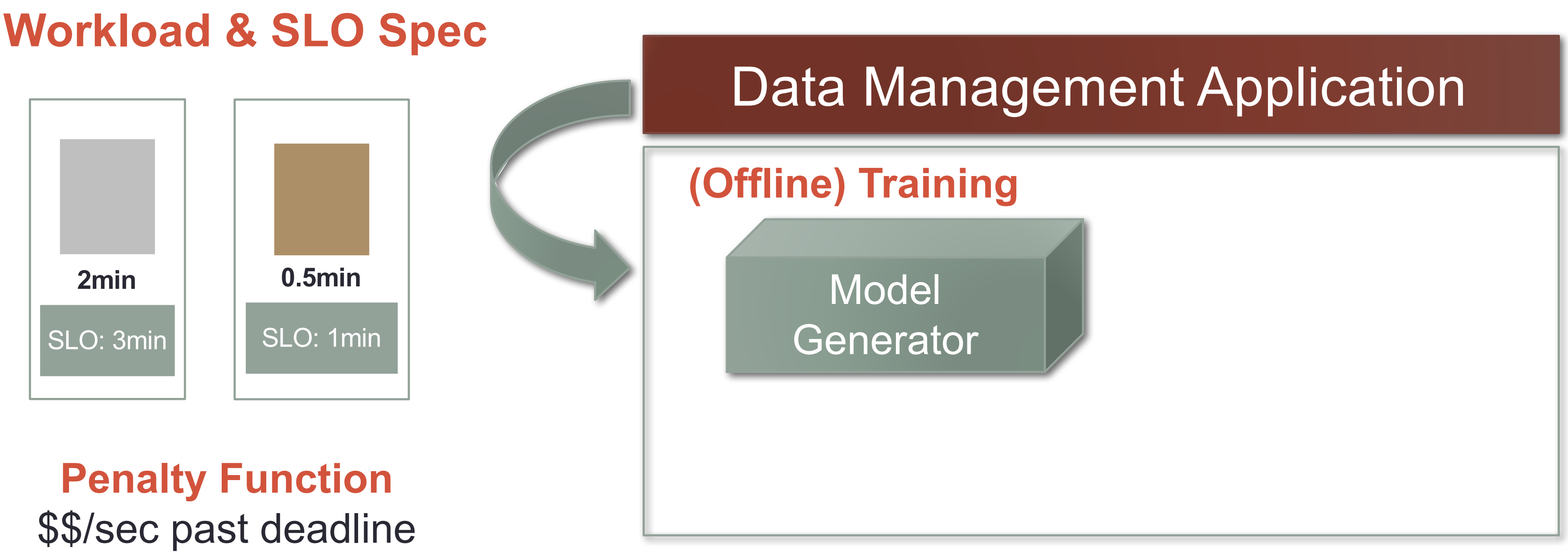


Outline

- Motivation
- **Offline Learning**
  - ❑ System Overview
  - ❑ Supervised Learning
  - ❑ Adaptive Learning
- Online Learning
- Conclusions

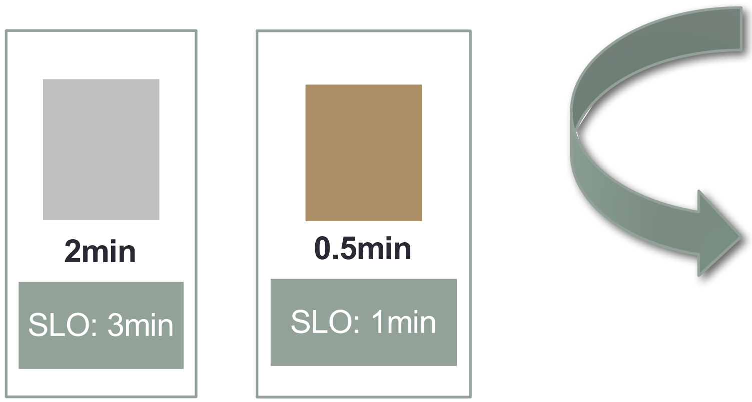
WiSeDB: A Learning-based Workload Management Advisor for Cloud Databases, Ryan Marcus, Olga Papaemmanouil, ***VLDB 2016***

WiSeDB – Batch Processing



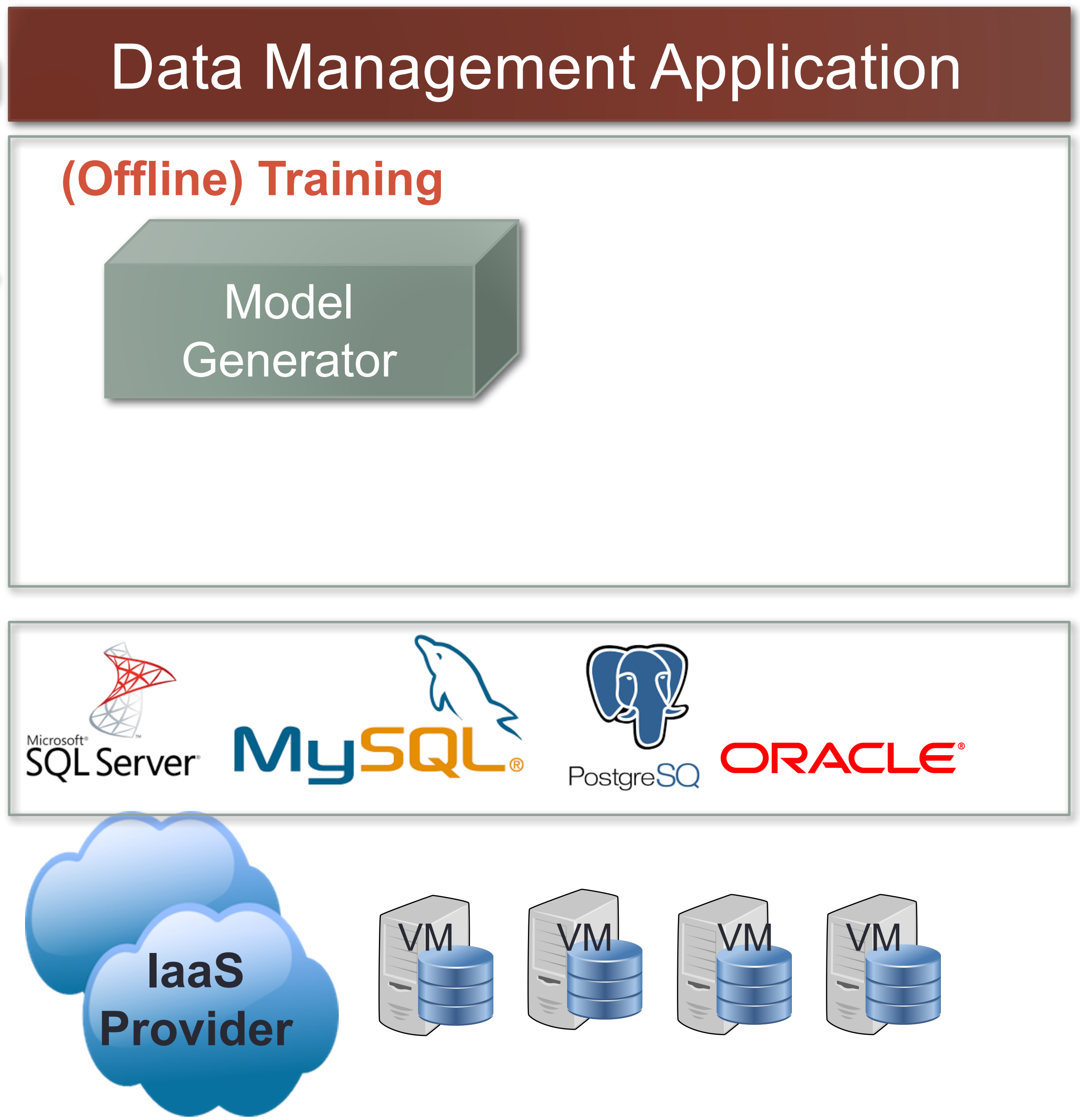
WiSeDB – Batch Processing

Workload & SLO Spec



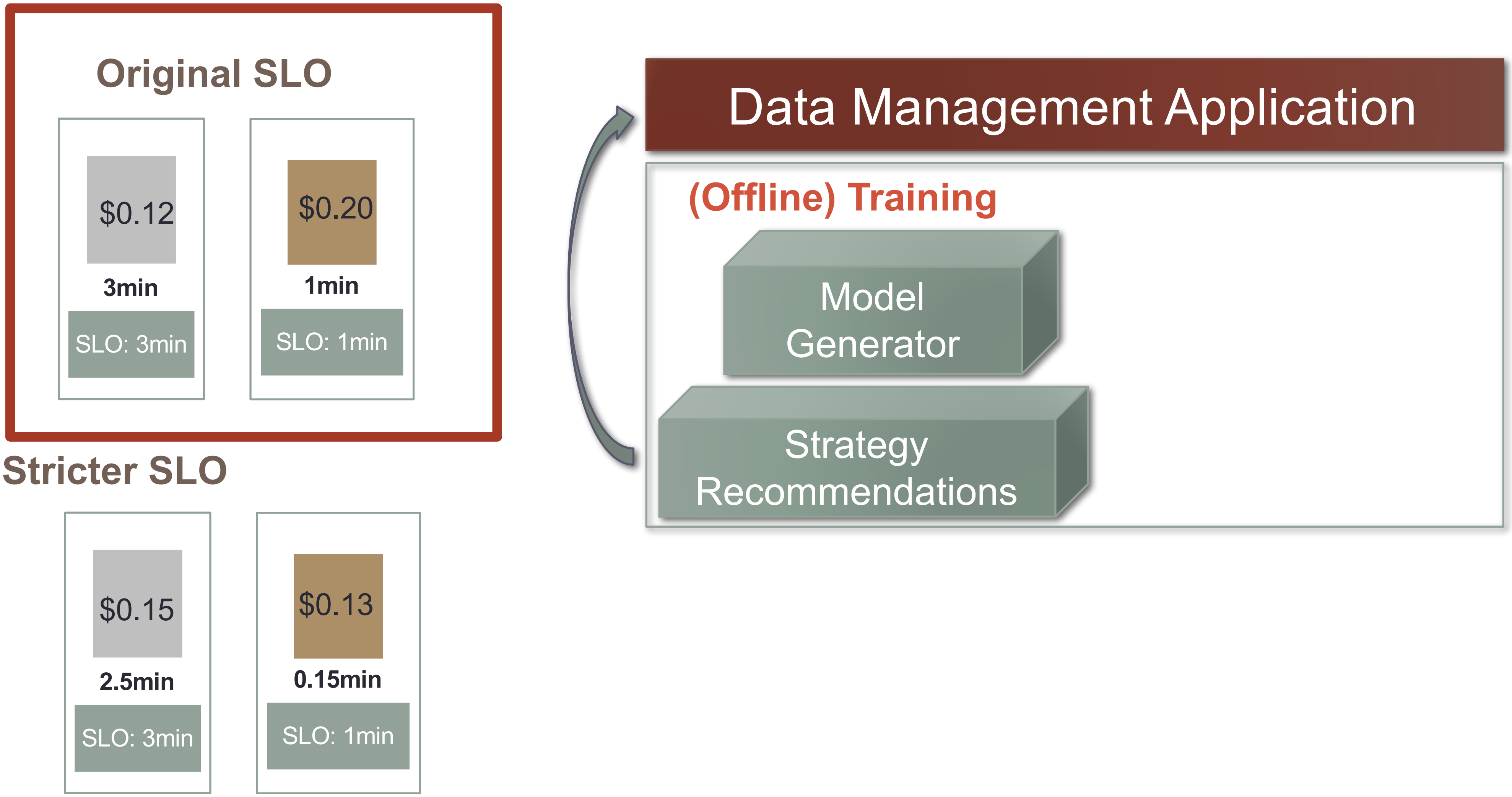
SLA Spec

$$/sec past deadline



- ❑ OLAP on full replicas (no updates)
- ❑ Known queries
- ❑ Performance model

WiSeDB – Batch Processing



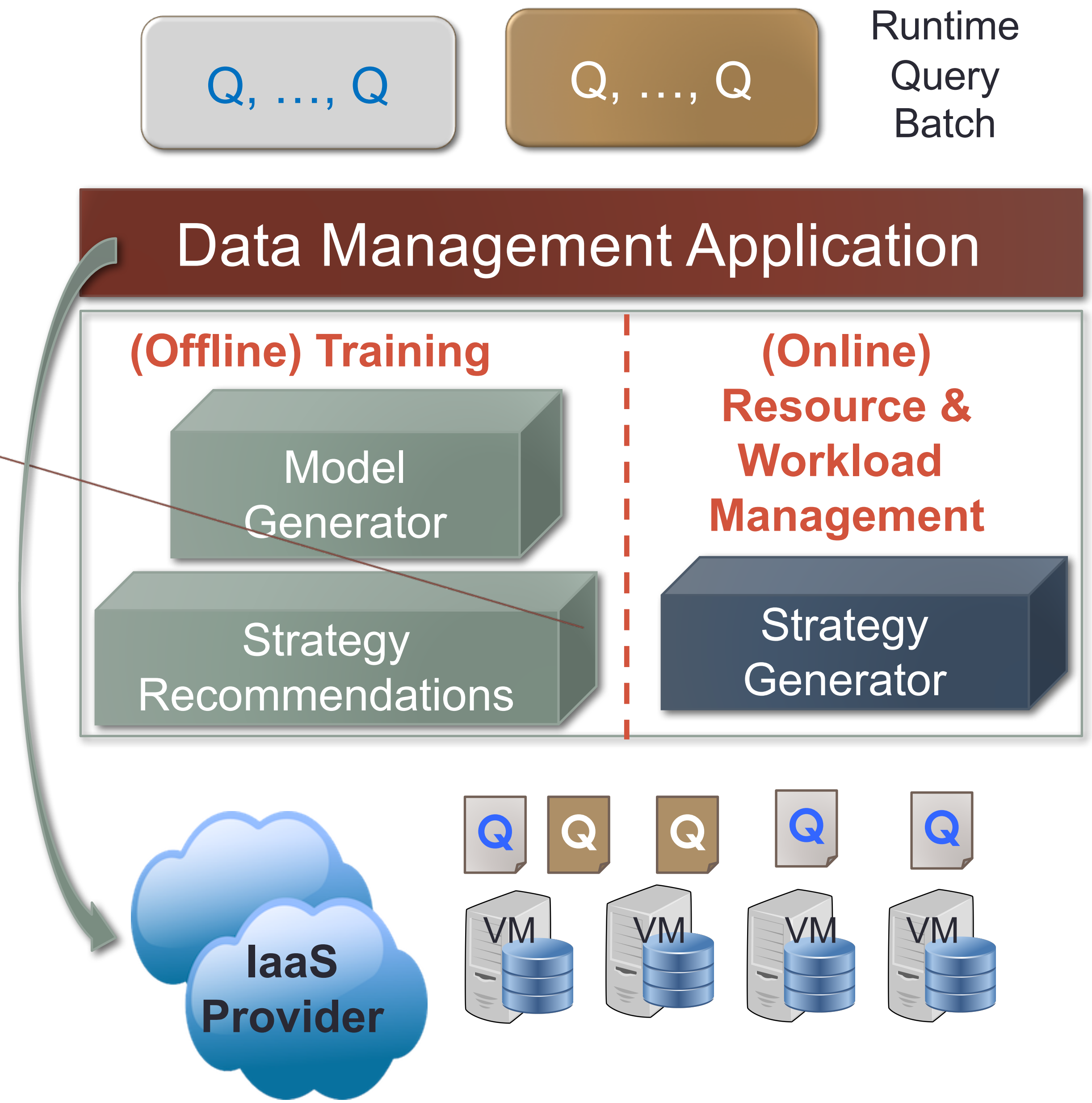
Batch Execution

Resources to rent

- ❑ # VMs/ type

Query scheduling

- ❑ Query execution order for each recommended VM



ASSUMPTIONS

- ❑ OLAP on full replicas (no updates)
- ❑ Known query types
- ❑ Performance prediction model

Supervised Learning

Model Generator

identify classes → classes == actions

- ❑ dispatch a query to a VM
- ❑ provision new VM

create training data → context of actions

- ❑ identify best decisions
- ❑ extract cost-related features

generate classifier → decision tree

- ❑ describe (context, action)
- ❑ interpretable: offers insight

“To be the best, learn from the best” (D. LaCroix)

Model Generator

Offline Learning

identify best decisions

1. Generate small workload
2. Build decision graph
   - query assignment
   - VM provisioning
3. Find optimal (minimum cost) solution (path)
4. Extract context of optimal decisions

generate model

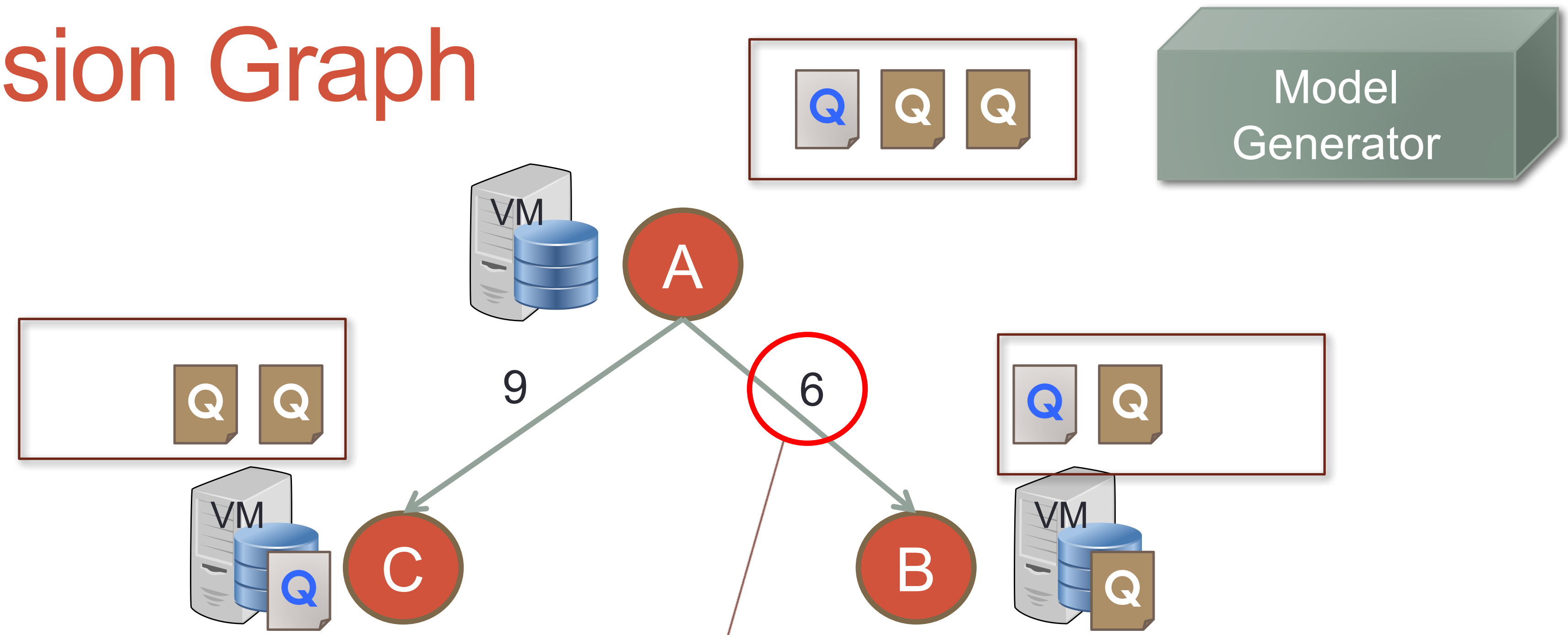
1. Repeat for many sample workloads
2. Build a training set of (feature, action)
3. Train a classifier

Runtime Scheduling

apply model

- Use classifier for
  - batch scheduling
  - online scheduling
  - performance vs cost exploration

Decision Graph

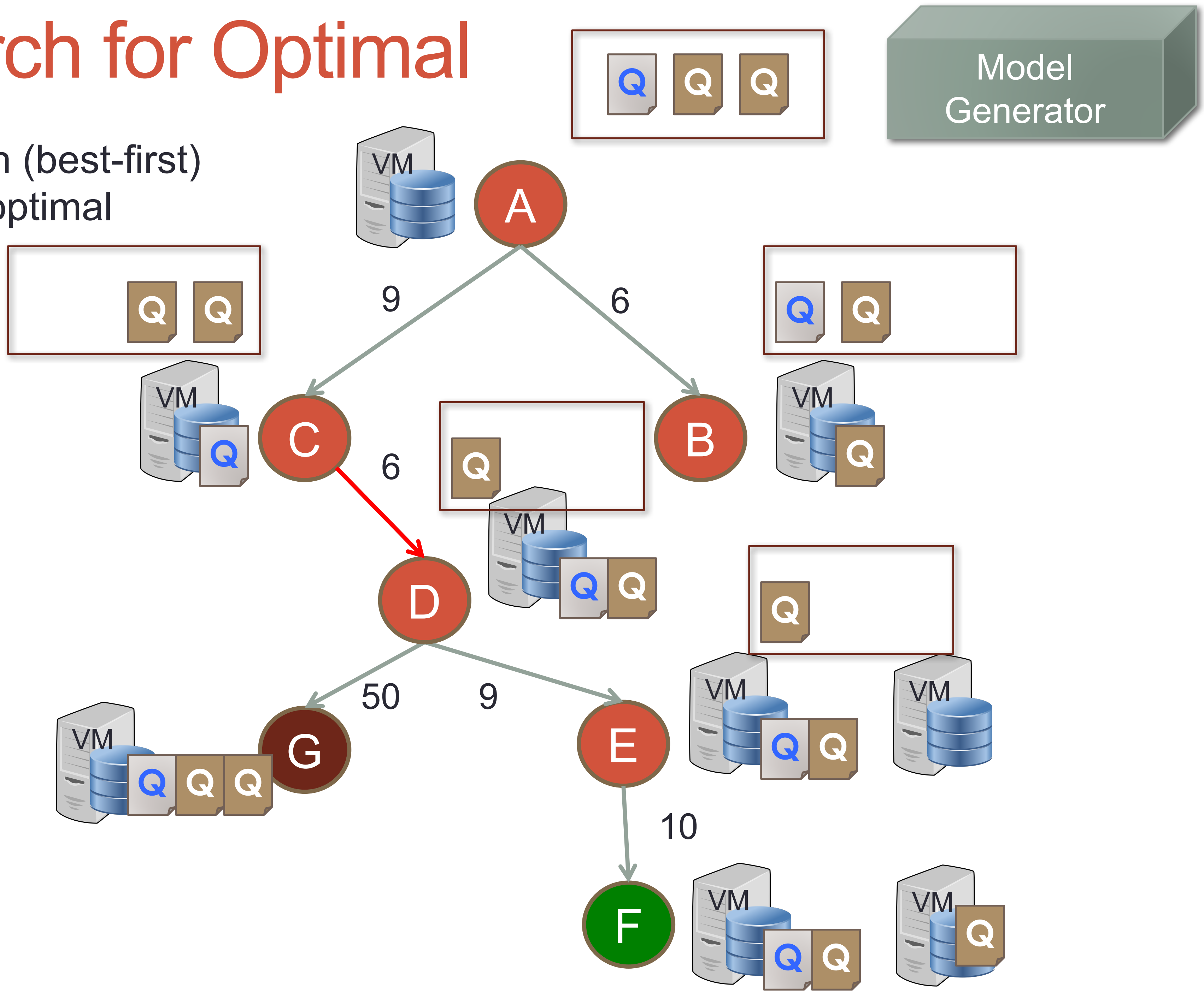


Monetary Cost

- Resource usage ($$/time)
  - time = VM start up + query execution
- Violation fees
  - Penalty function (black box)

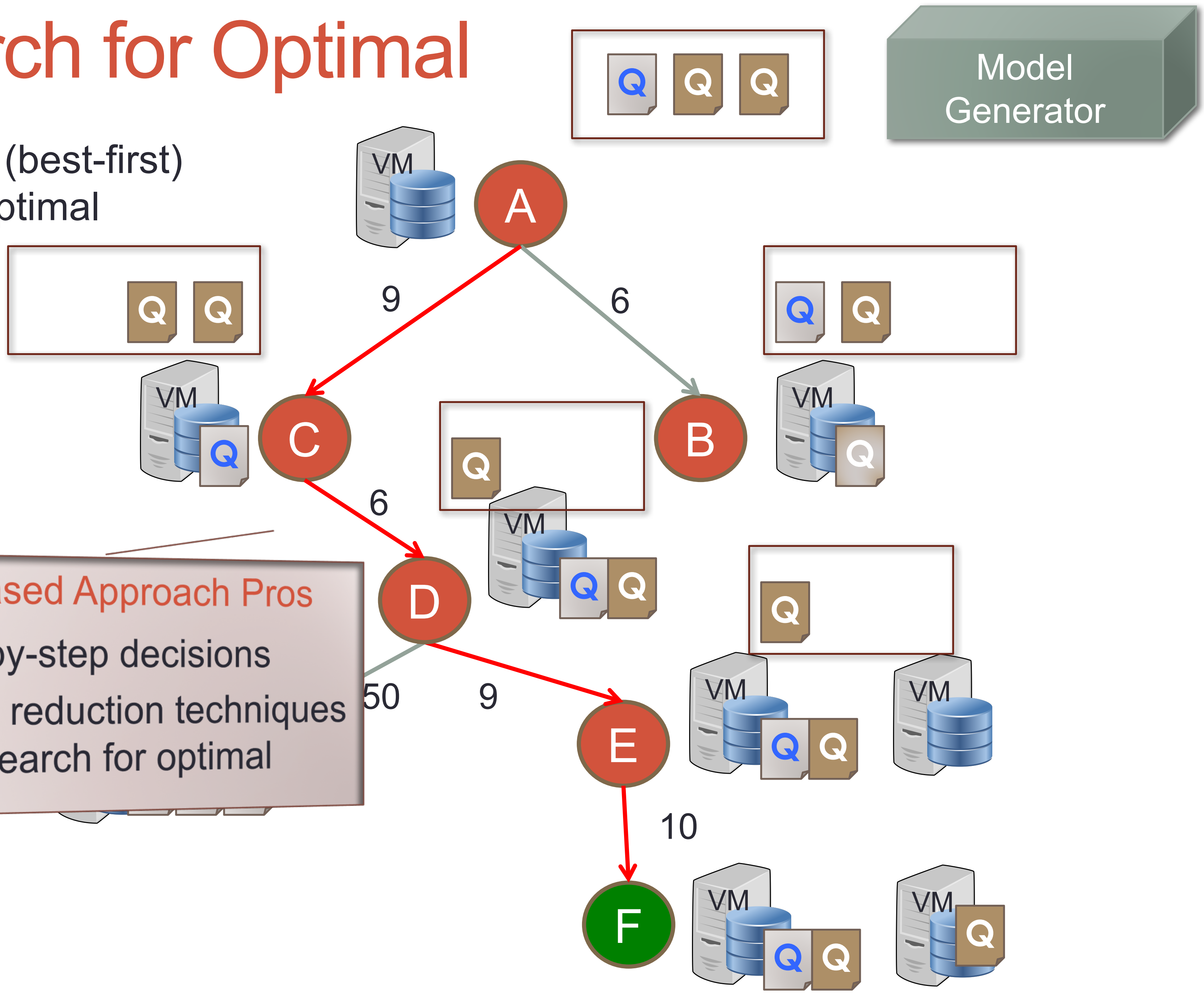
Search for Optimal

A* search (best-first)
for optimal



Search for Optimal

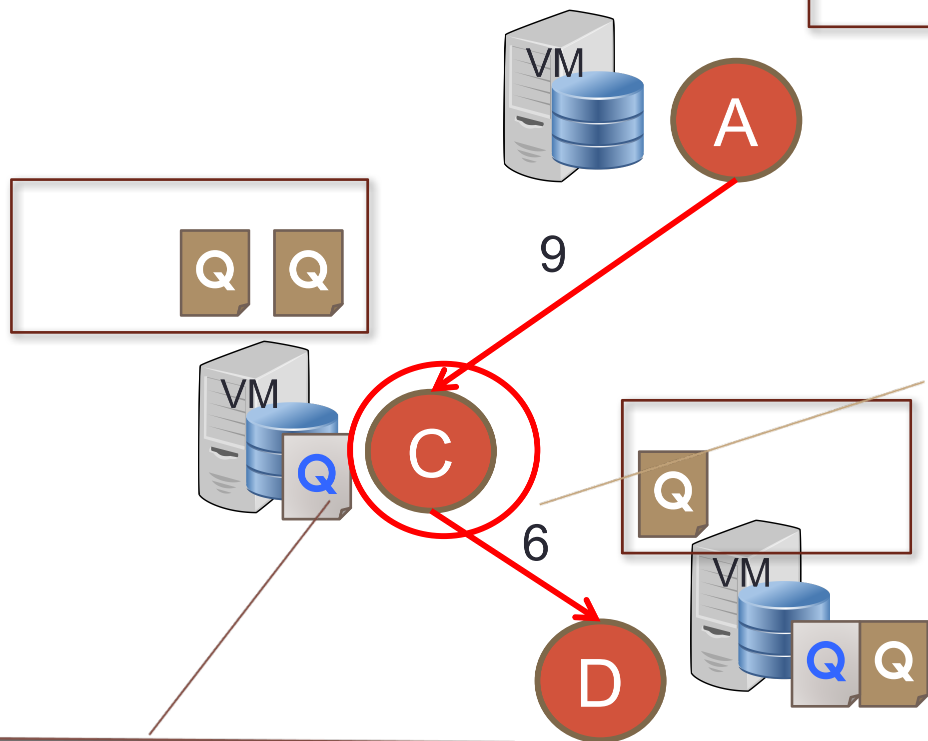
A* search (best-first)
for optimal



Graph-based Approach Pros

- Step-by-step decisions
- Graph reduction techniques
- Fast search for optimal

Feature Extraction



Decision: Assign Q to VM

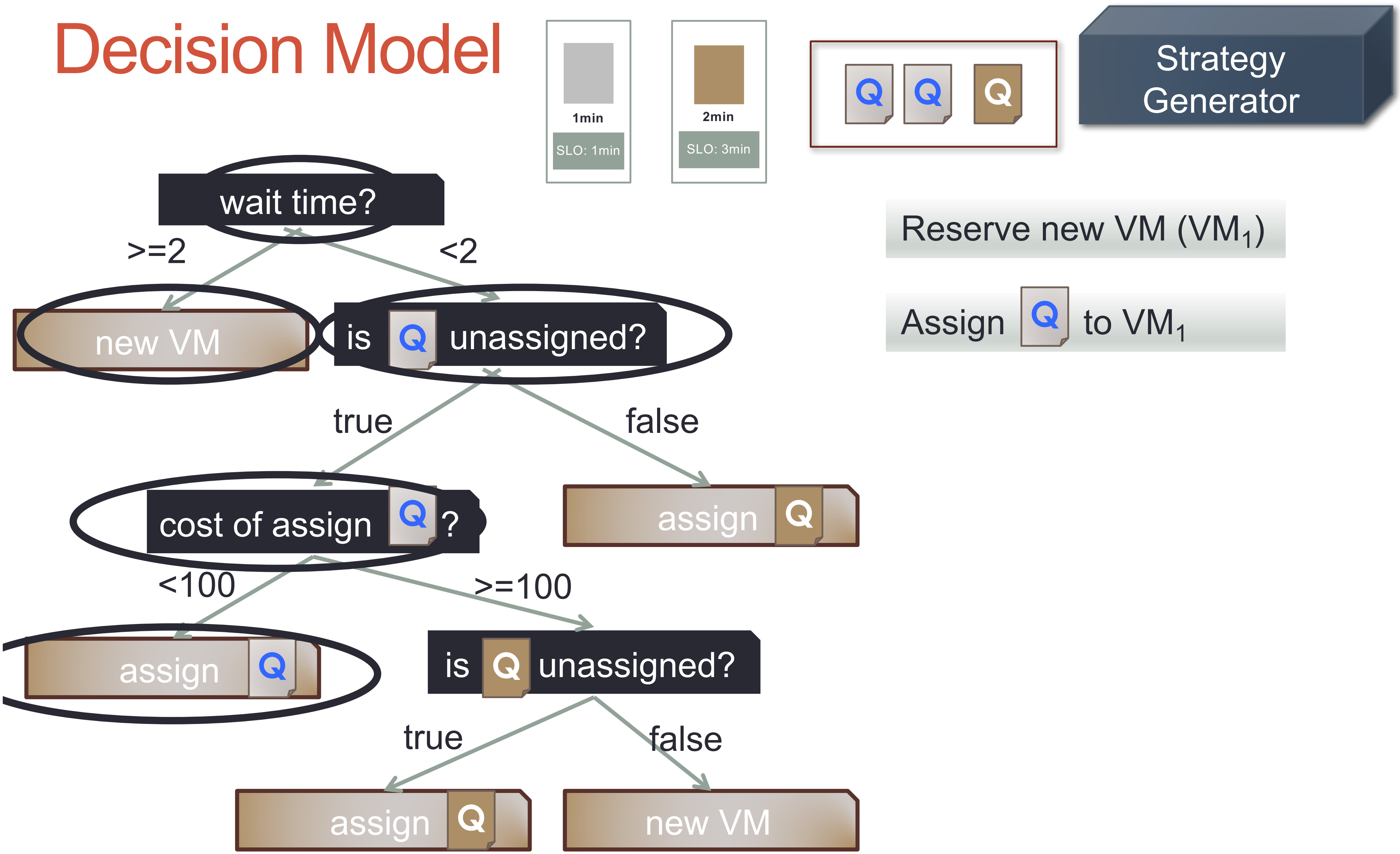
Features:

- unassigned Q : true
- unassigned Q : false
- cost of assigning Q :$2
- wait time on VM: 1min
- % of Q in VM: 50%
- % of Q in VM: 50%

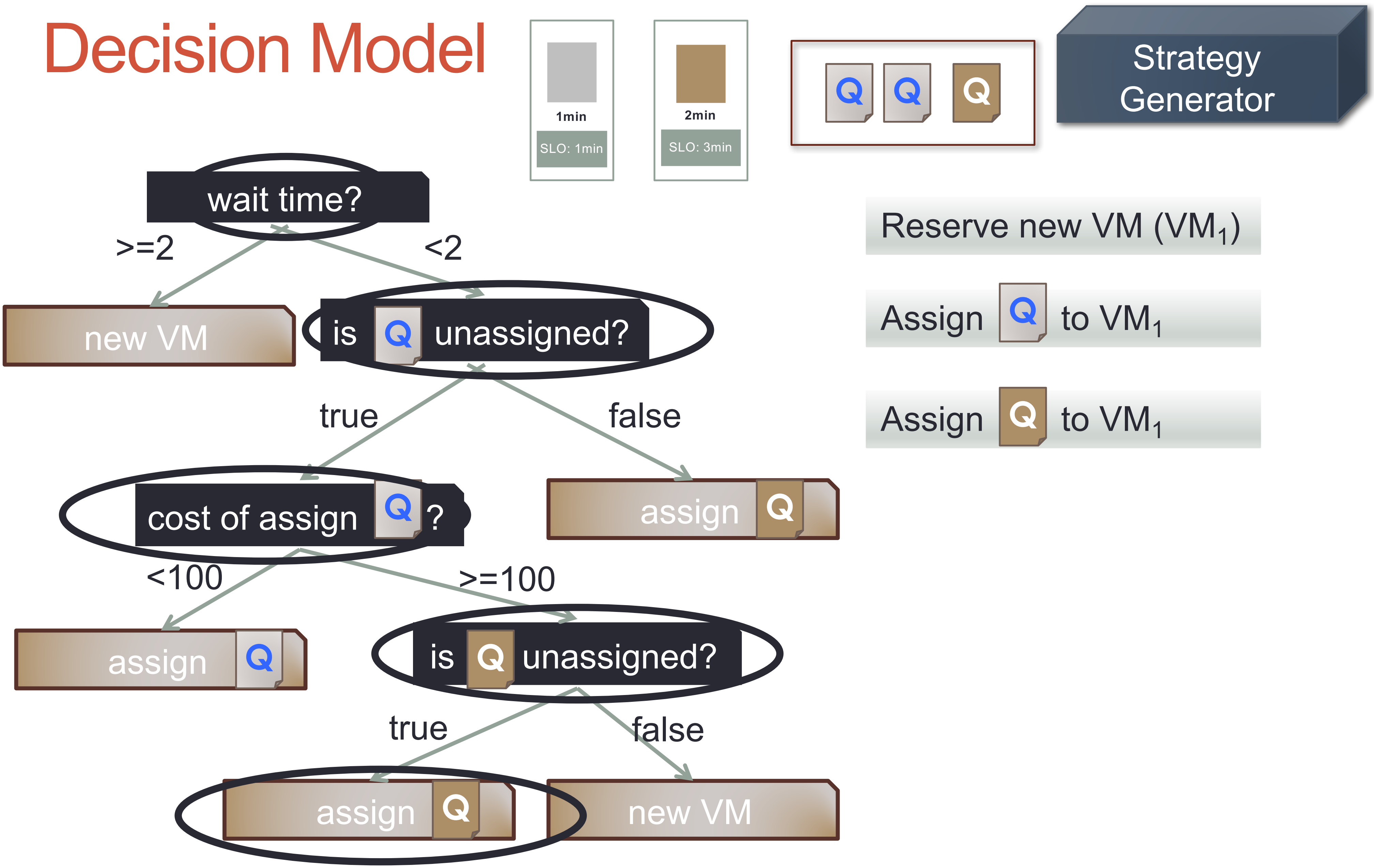
Agnostic to

- Query semantics
- Performance goal (SLO)
- Workload size

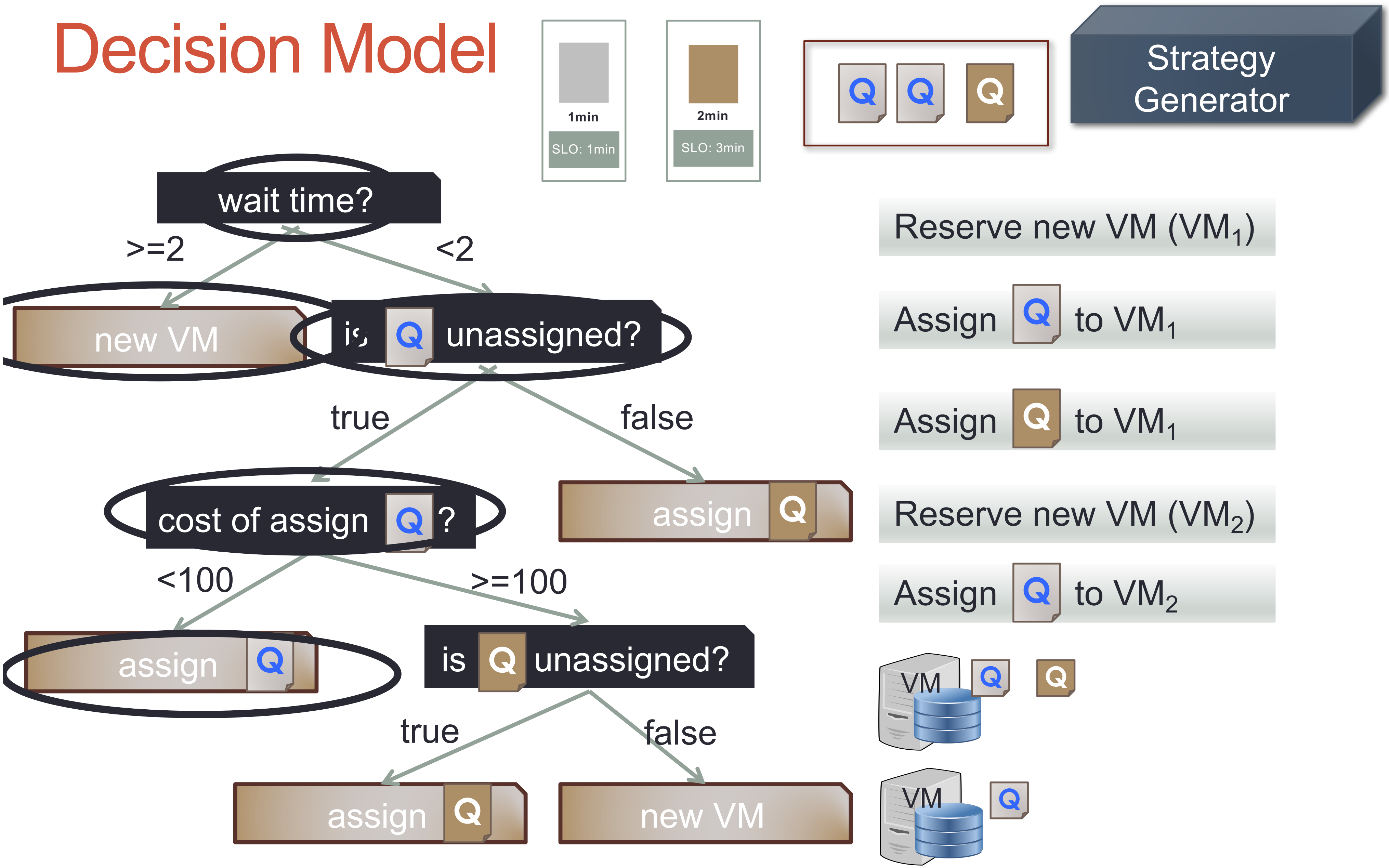
Decision Model



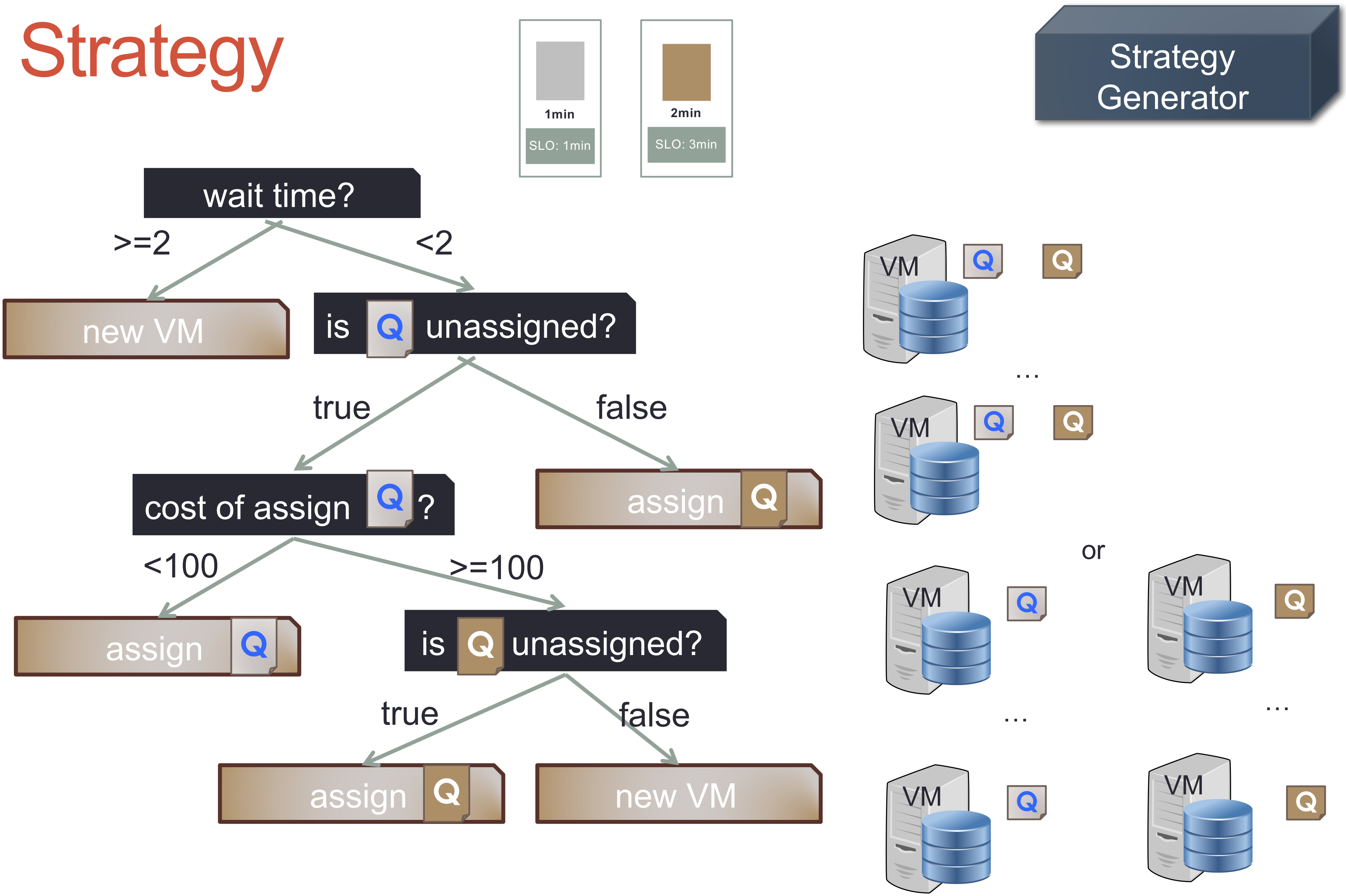
Decision Model



Decision Model



Strategy



1min
SLO: 1min
2min
SLO: 3min
Strategy Generator
wait time?
>=2
<2
new VM
is Q unassigned?
true
false
cost of assign Q ?
assign Q
<100
>=100
assign Q
is Q unassigned?
true
false
assign Q
new VM
VM
Q
Q
...
or

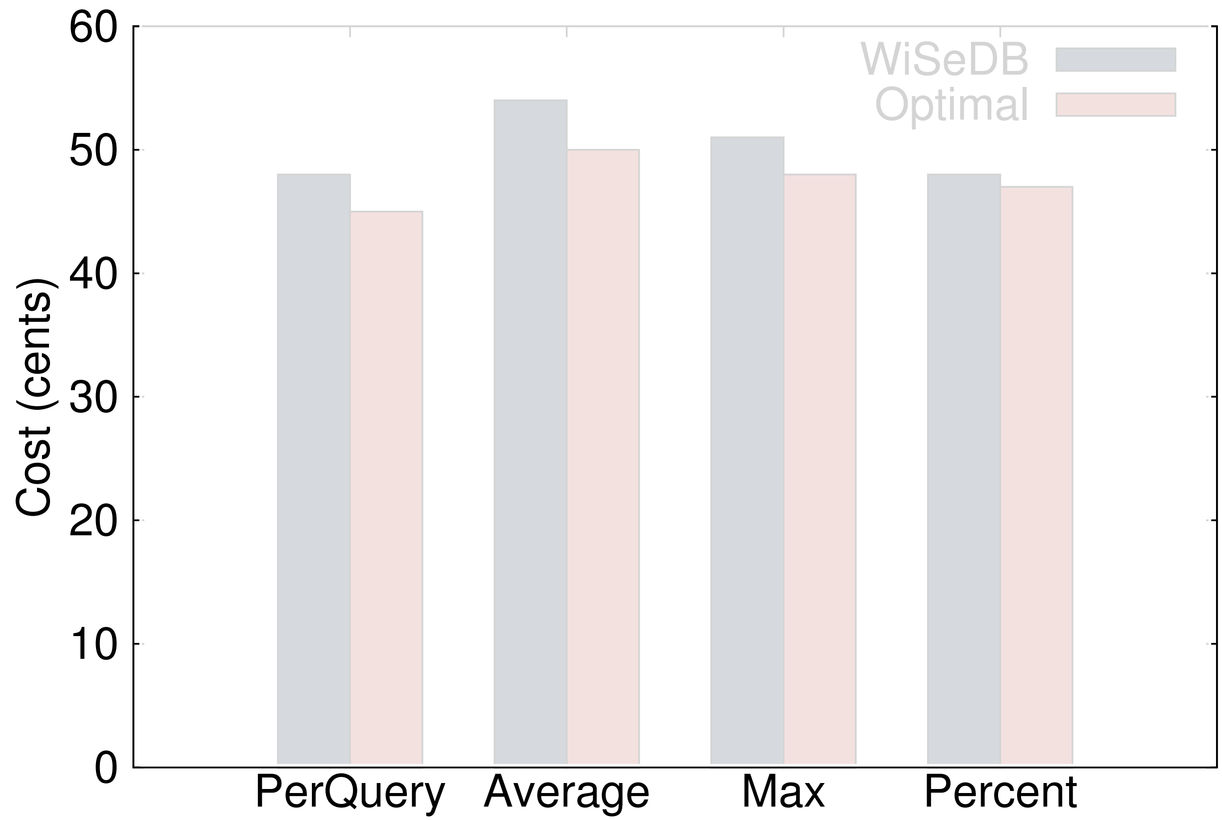
Experimental Setup

Training Data

3000 samples

10 TPC-H templates

18 queries/sample



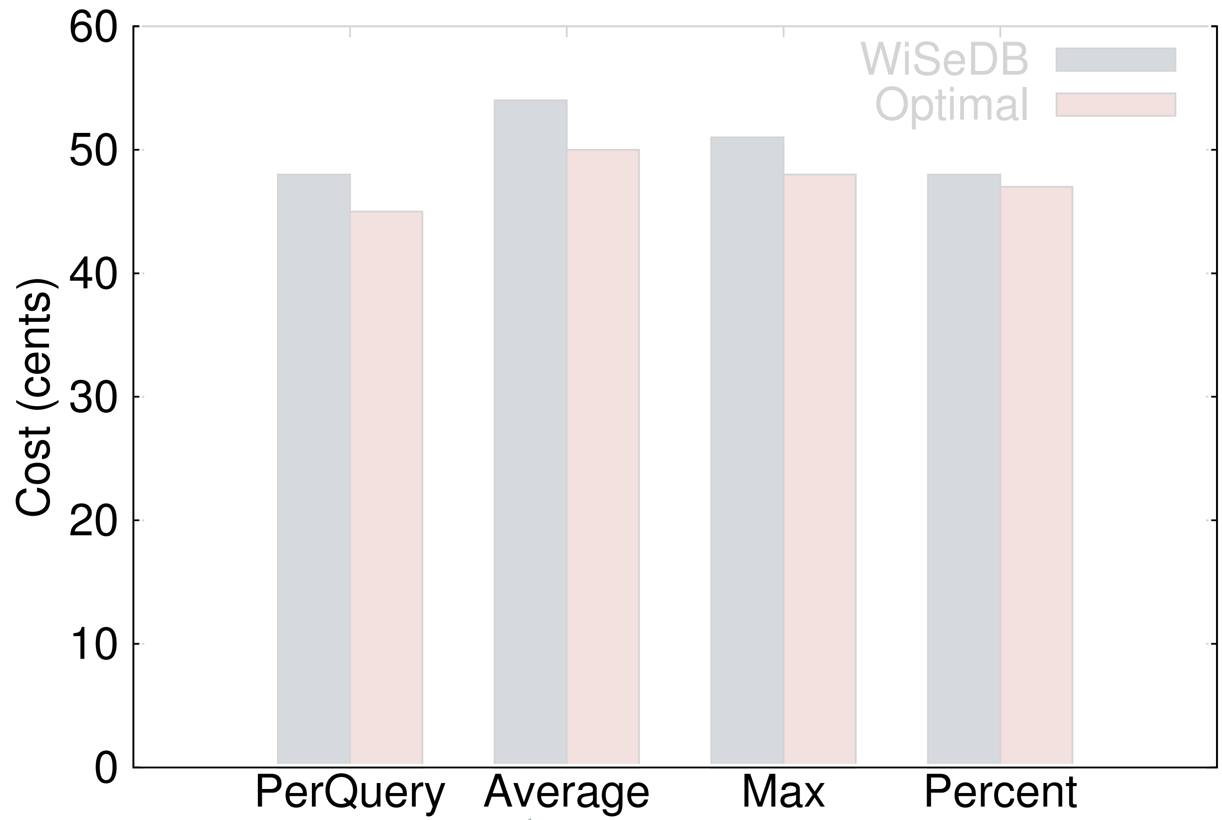
query execution time <=x secs
(same deadline per template)

Experimental Setup

<u>Training Data</u>

3000 samples

10 TPC-H templates
18 queries/sample



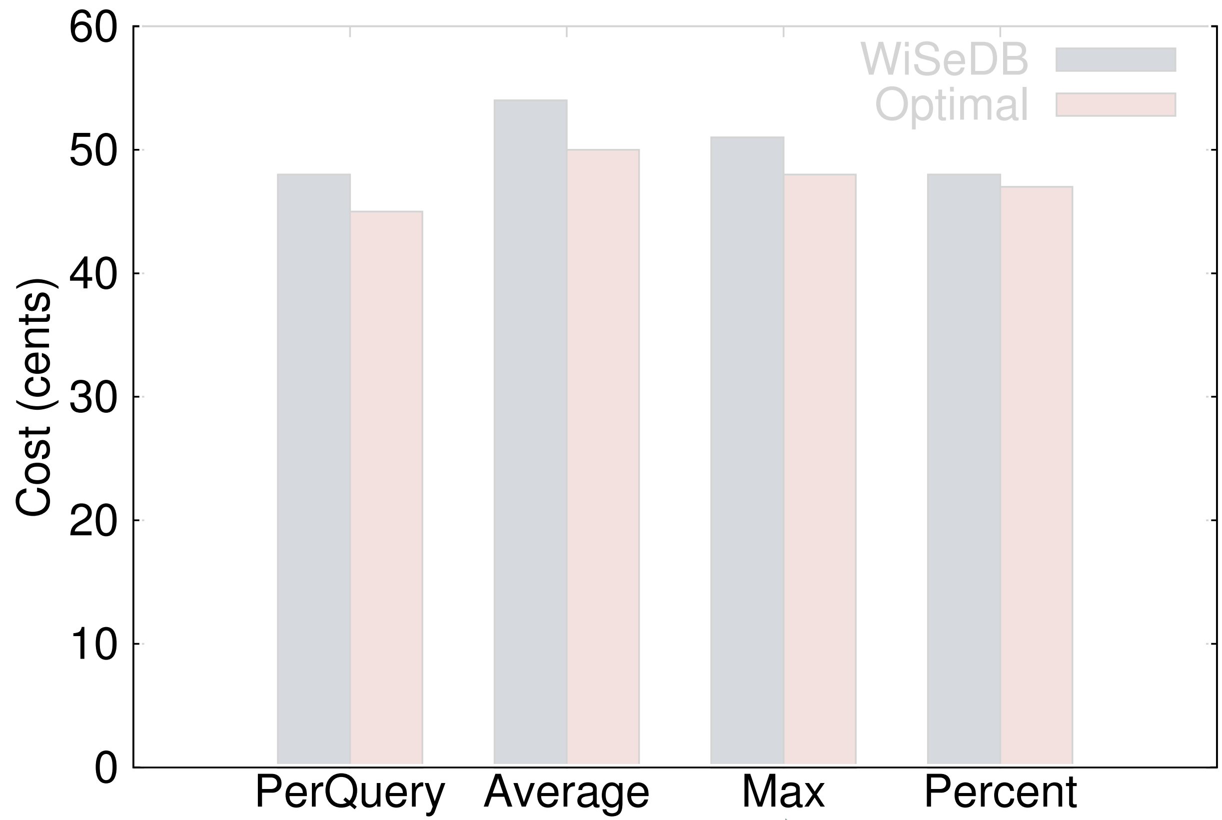
average latency of the workload <= x secs

Experimental Setup

Training Data

3000 samples

10 TPC-H templates
18 queries/sample



max latency <=x secs
(longest query in the workload)

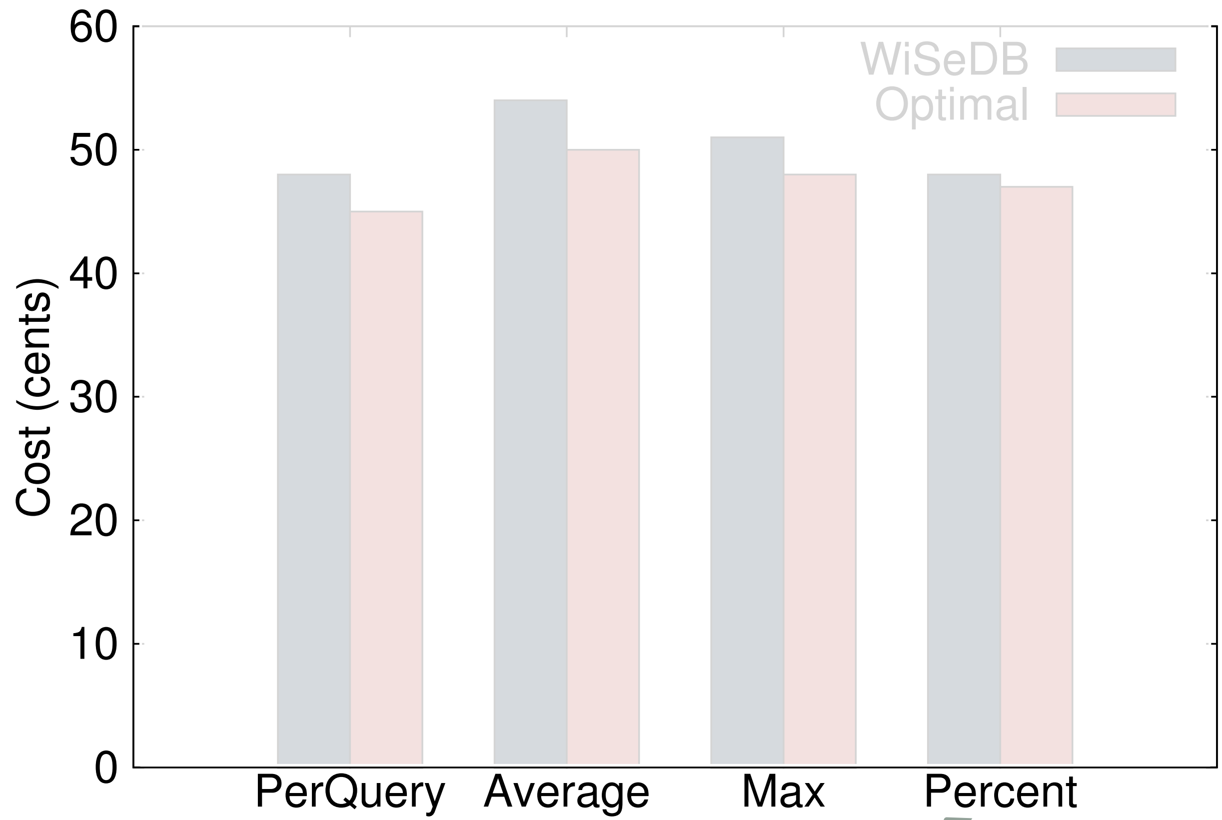
Experimental Setup

Training Data

3000 samples

10 TPC-H templates

18 queries/sample



execution time of 90% of queries
in the workload <= x secs

Experimental Setup

Training Data

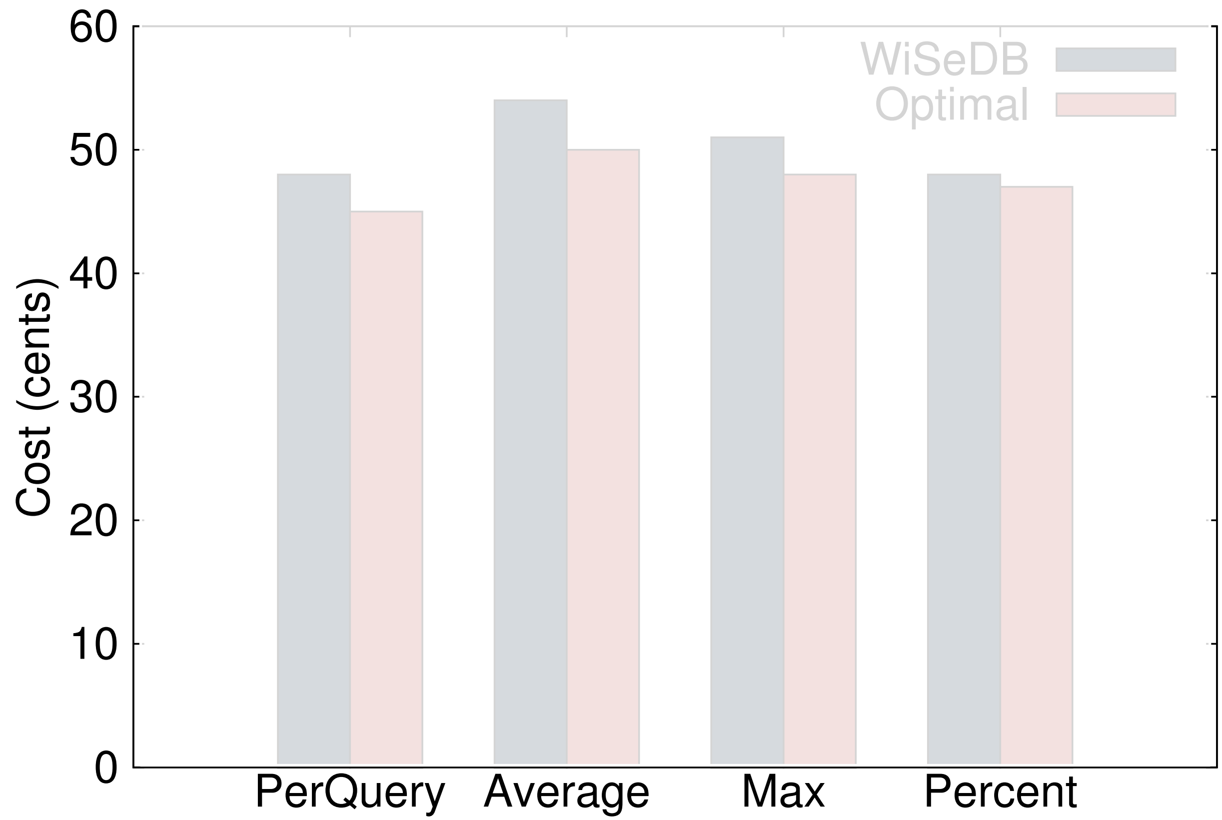
3000 samples

10 TPC-H templates
18 queries/sample

Testing Data

10 TPC-H templates

varied queries/workload



Experimental Setup

Training Data

3000 samples

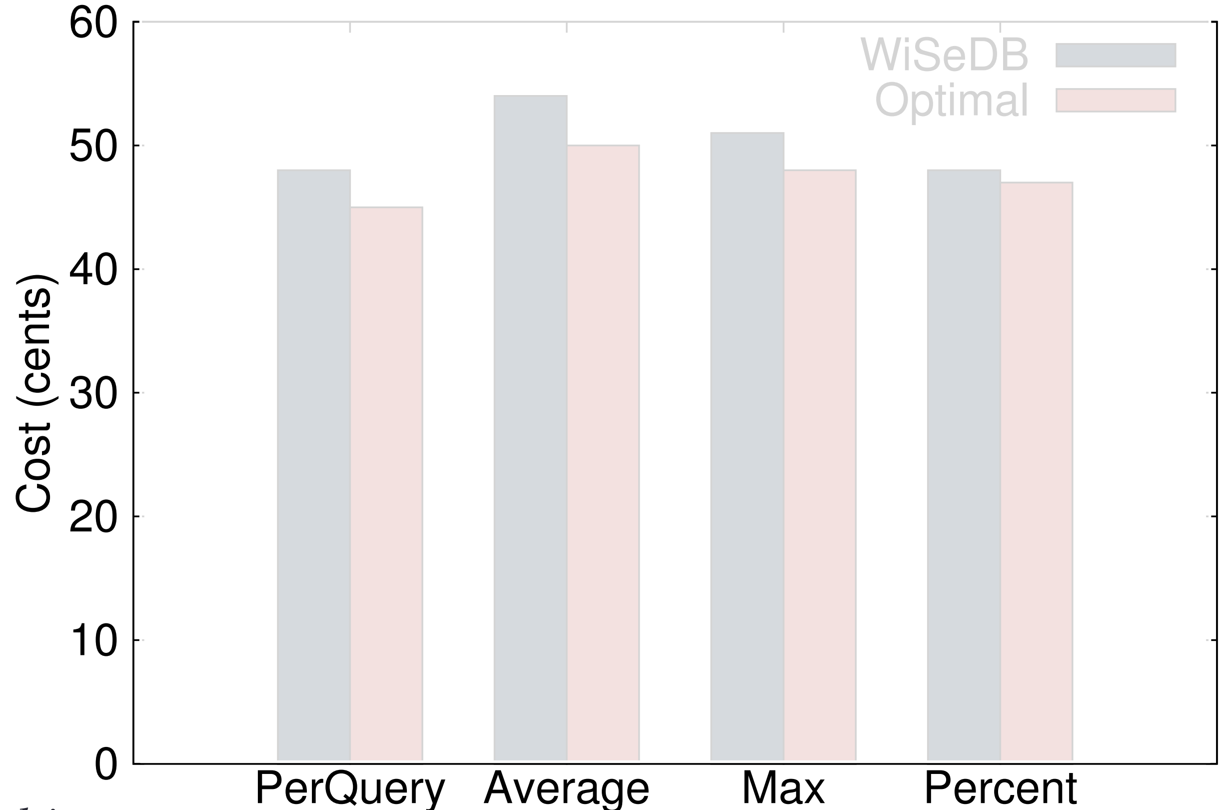
10 TPC-H templates

18 queries/sample

Testing Data

10 TPC-H templates

varied queries/workload



cost: resource utilization+ penalties

AWS Cloud

fees penalty $0.01/sec of violation

Effectiveness (small workloads)

Training Data

3000 samples

10 TPC-H templates

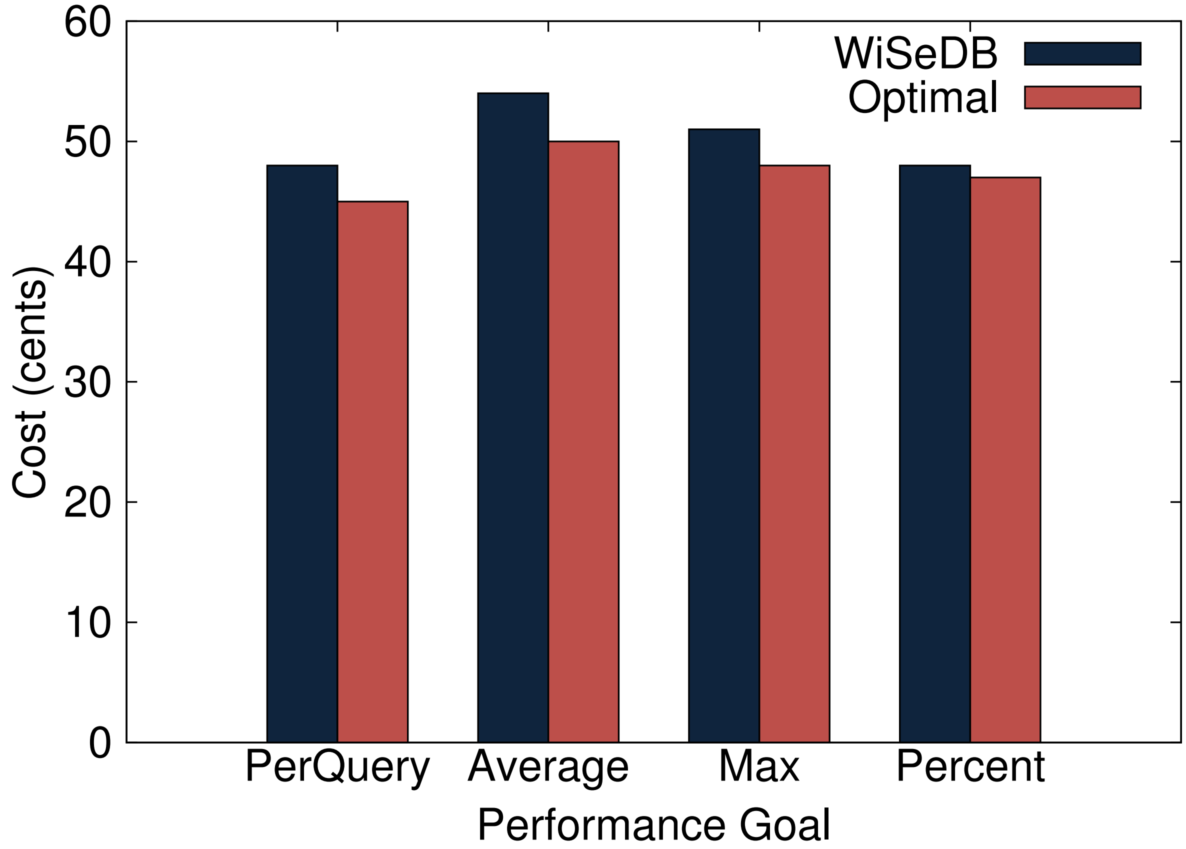
18 queries/sample

Testing Data

10 TPC-H templates

30 queries/workload

***Optimal*: Brute force**



WiSeDB models are within 8% of the minimum cost solution

Effectiveness (large workloads)

Training Data

3000 samples

10 TPC-H templates

18 queries/sample

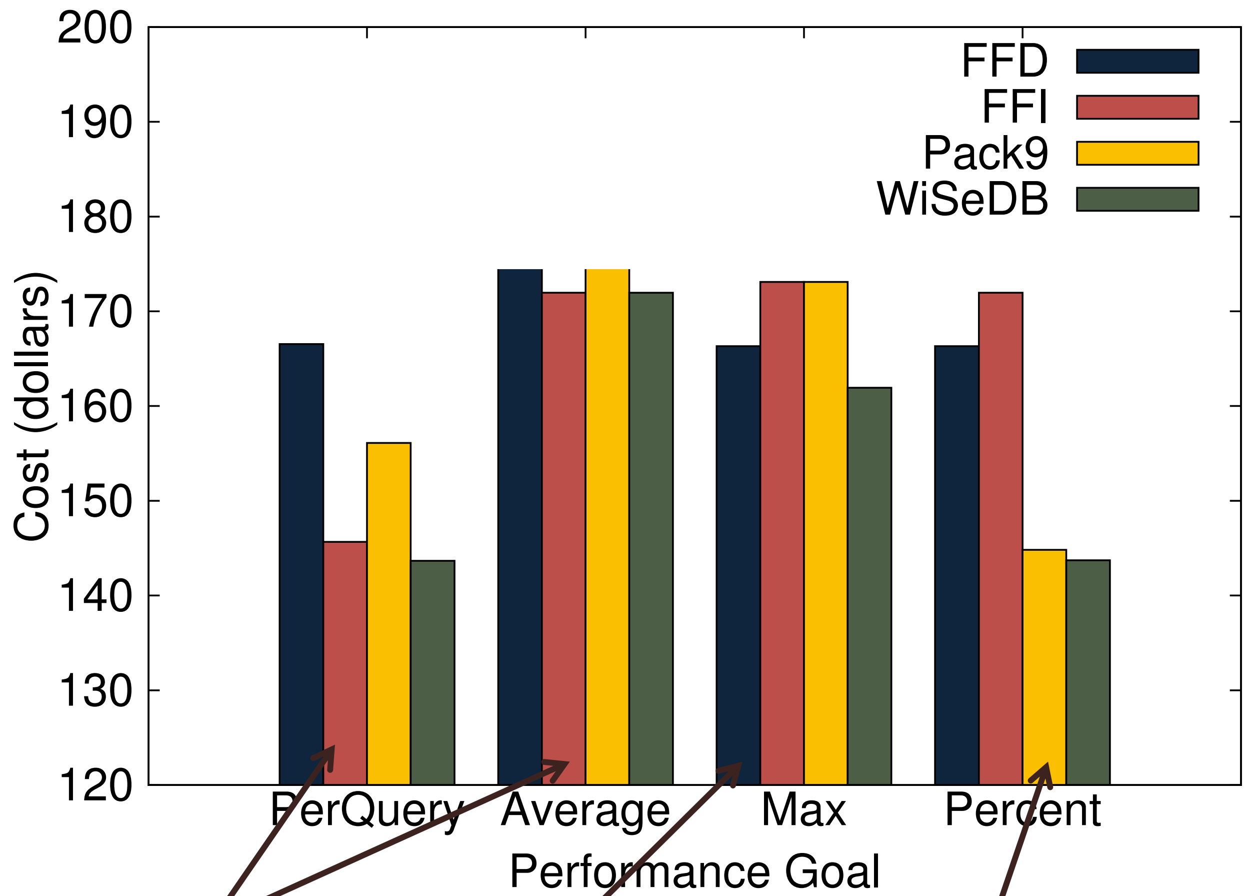
Testing Data

10 TPC-H templates

5000 queries/workload

One heuristic cannot fit all

WiSeDB learns the right heuristic



Best: shortest query first

Best: longest query first

Best: top-90% shortest then 10% longest queries

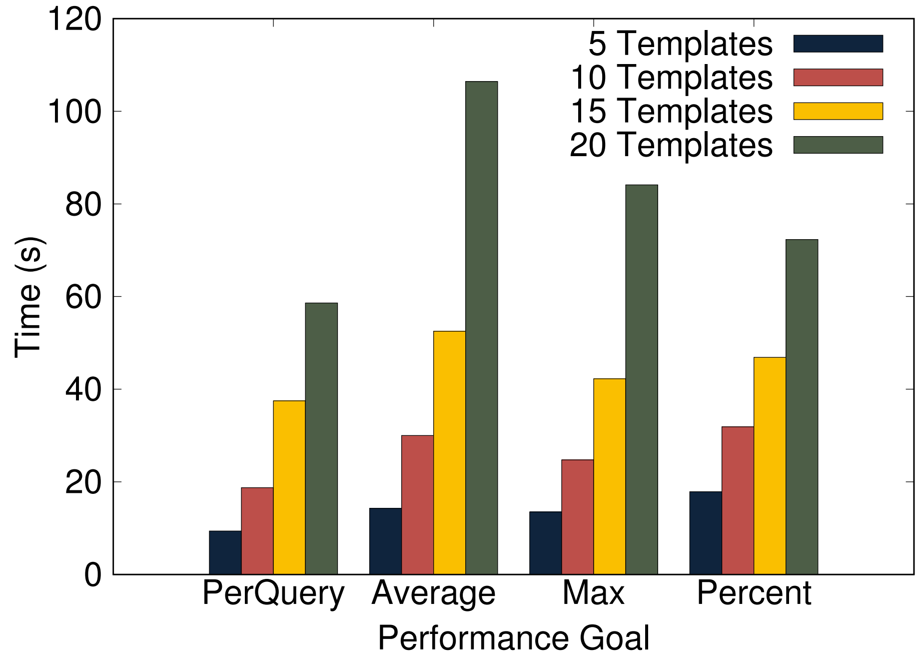
Training Overhead

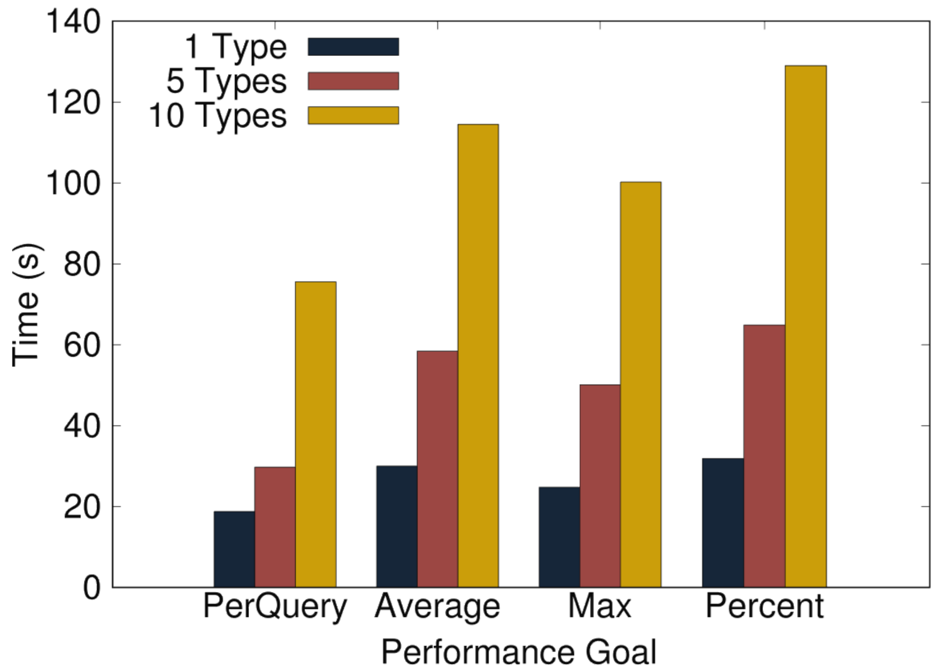
Training Data

3000 samples

10 TPC-H templates

18 queries/sample





Offline learning overhead 20sec – 120 sec

Beyond Batch Scheduling

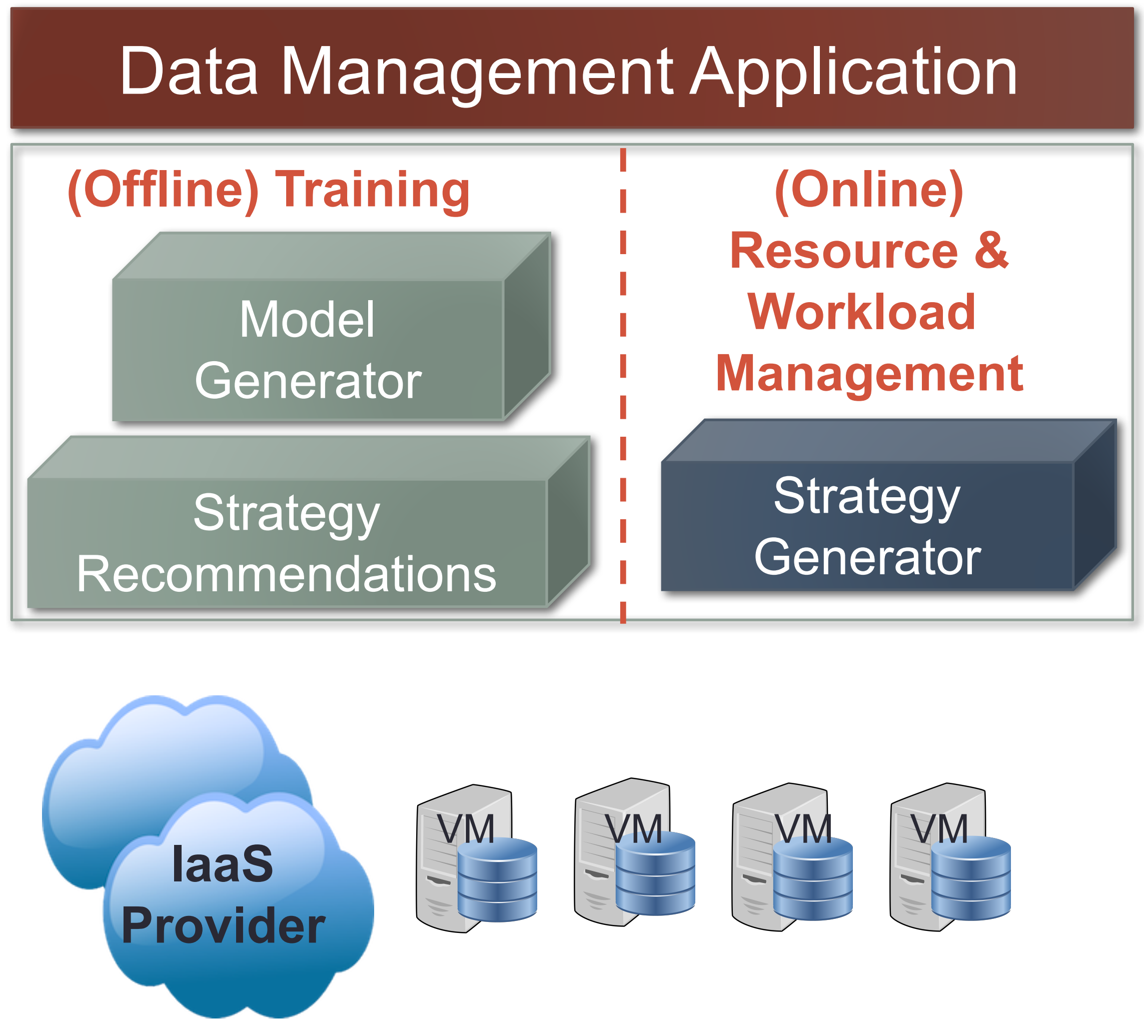
- Efficient performance vs cost trade off exploration
  - Recommend strategies with stricter/looser performance goals
  - Reuse original training set to generate **quickly** alternative models
    - Best-first heuristic reduces search time (dominant training factor)
  - Training overhead improvement by **96-98%**

- Online scheduling (query at a time)
  - Challenge: arrival times are unknown and hence not modeled
  - Generate a new model upon arrival of new query: too expensive
  - Optimization 1: Adapt previous model to reduce training overhead
  - Optimization 2: Reuse past models, when feasible

Offline Learning

Advantages

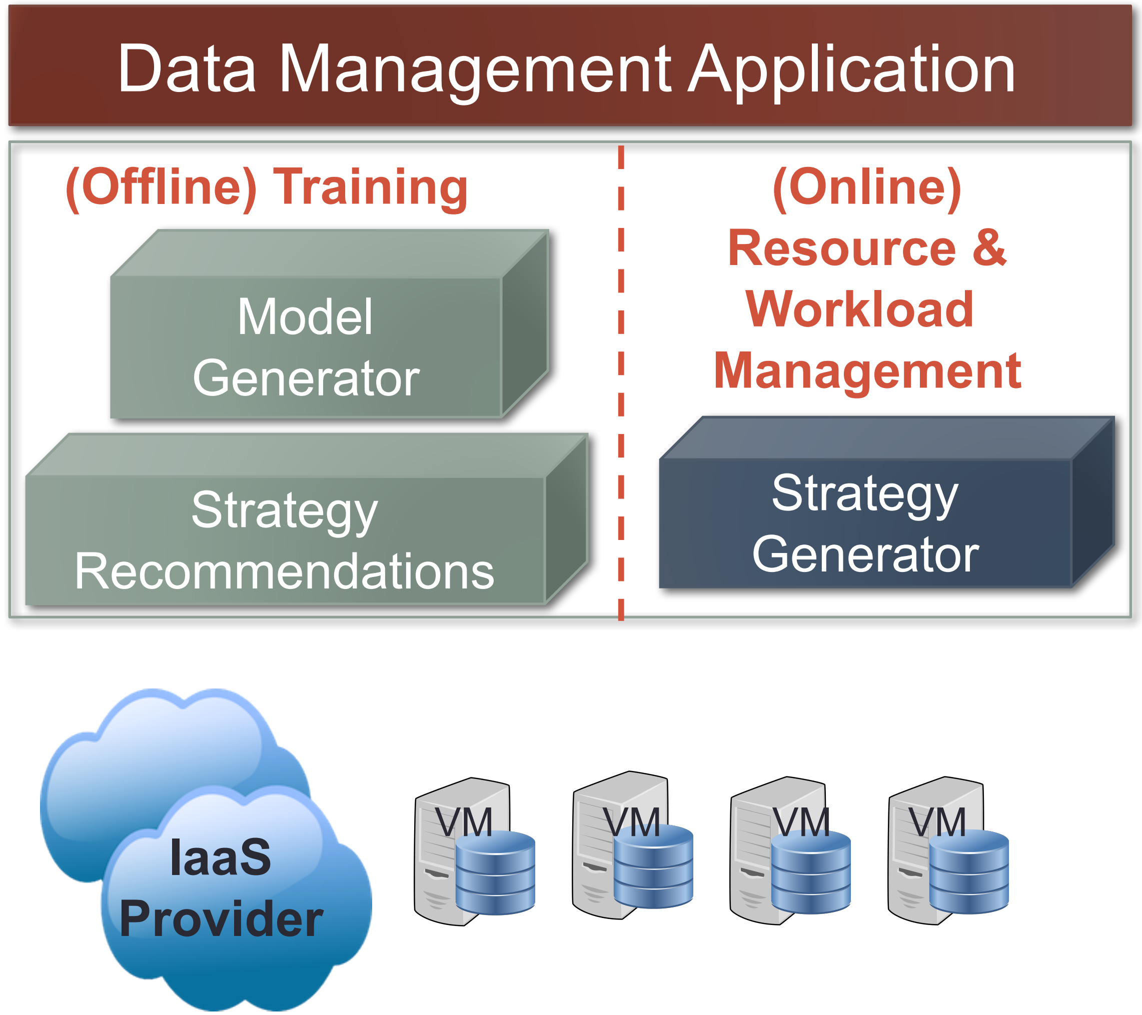
- Provides insight on complex decisions
- Learns custom strategies per application
- Explores performance vs cost trade-offs



Offline Learning

Limitations

- Static decision models
- Batch scheduling
- Performance model



Outline

- Motivation
- Offline Learning
- Online Learning
- Conclusions

❑Explicit vs Implicit Modeling

❑Reinforcement Learning

Releasing Cloud Databases from the Chains of Predictions Models.
Ryan Marcus, Olga Papaemmanouil, ***CIDR 2017***

(Explicit) Performance Prediction

- DBMS-related challenges
  - isolated vs. concurrent query execution
  - low accuracy for new query types (“templates”)
  - extensive off-line training
  - state-of-the-art: 15-20% prediction error*

- Cloud-related challenges
  - “noisy neighbors”
  - numerous resource configurations
  - predictions errors accumulation

* *Contender: A Resource Modeling Approach for Concurrent Query Performance Prediction*,
Jenny Duggan, Olga Papaemmanouil, Ugur Cetintemel, Eli Upfal, **EDBT 2015**

* *Performance Prediction for Concurrent Database Workloads*,
Jennie Rogers, Ugur Cetintemel, Olga Papaemmanouil, Eli Upfal, **SIGMOD 2011**

WiSeDB: Implicit Performance Modeling

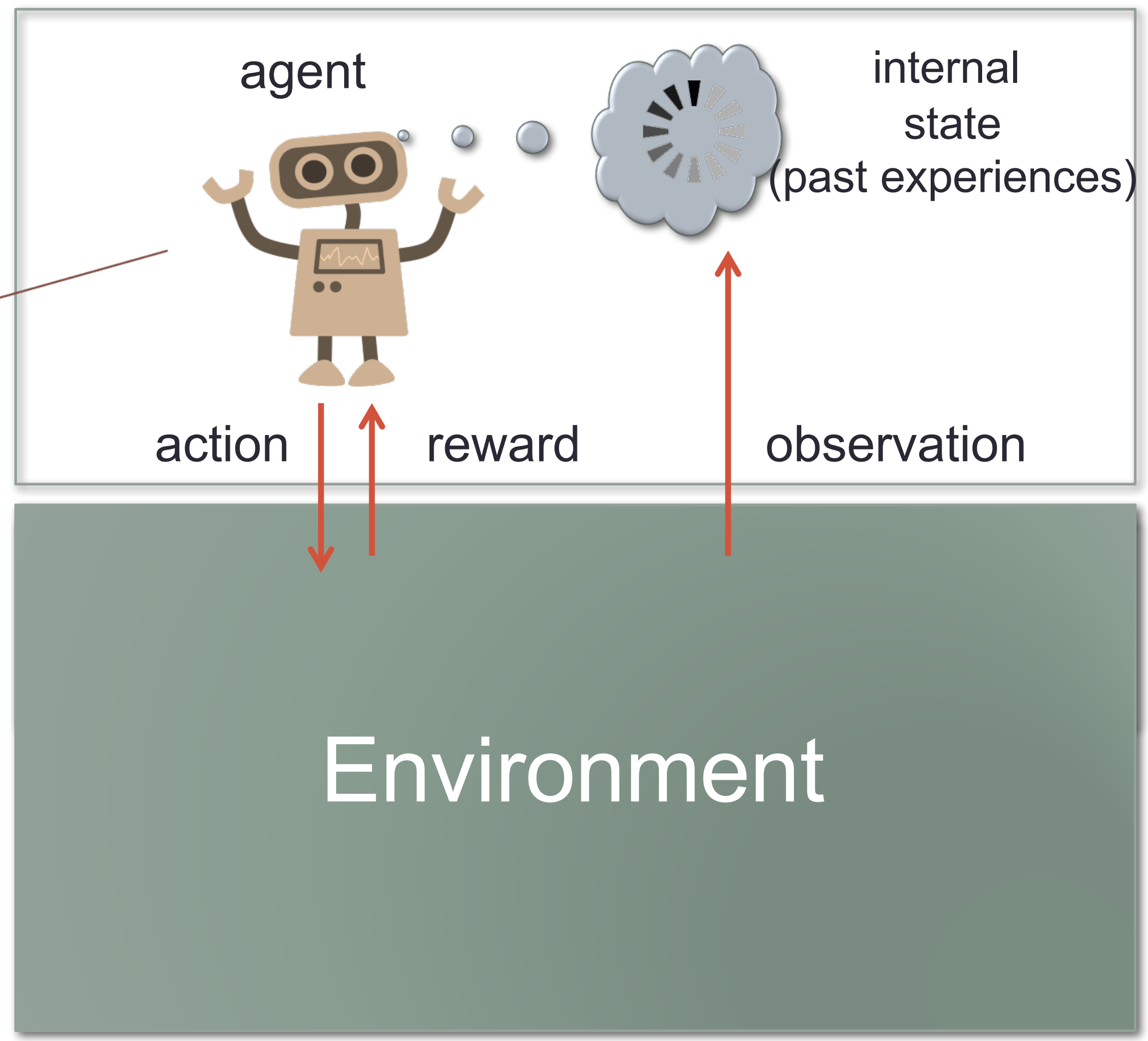
- Explicit performance models are NOT necessary for:
  - monetary cost management
  - resource & workload management
  - offer performance SLA and **keep penalties low**

- Implicitly model query latency
  - predict *monetary cost (& violation penalties)*
- Online training for dynamic environments
  - Automatic scaling & workload distribution

Wish List #2

Reinforcement Learning

- Continuous learning
- Explicit reward modeling
- Action selection
  - maximize reward



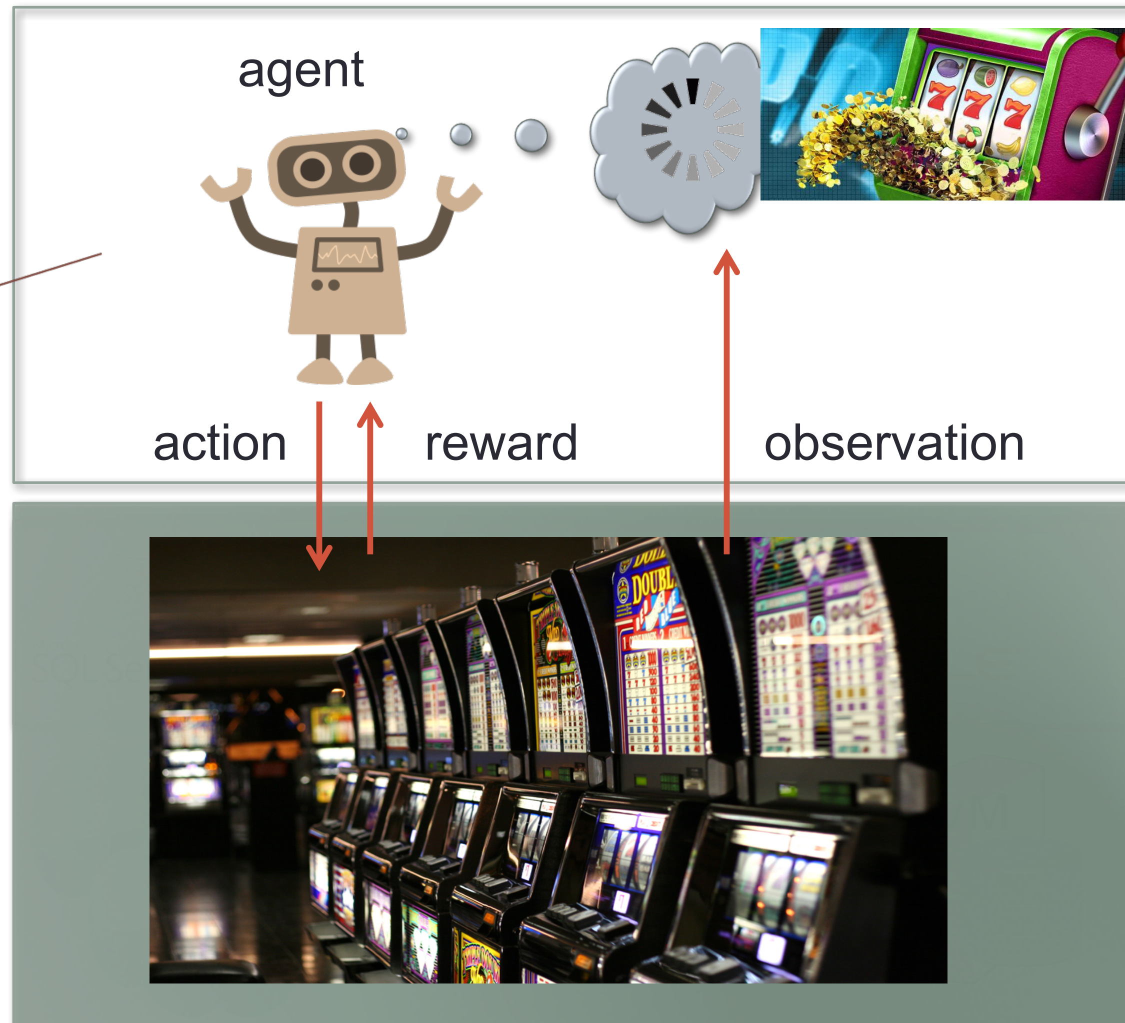
CMABs

(Contextual Multi-Armed Bandits)

Contextual Multi-Armed Bandit Problem

Armed Bandit = Slot Machine

*Which slot machine to play (**action**) so that you walk out with the most $$$ (**reward**)?*



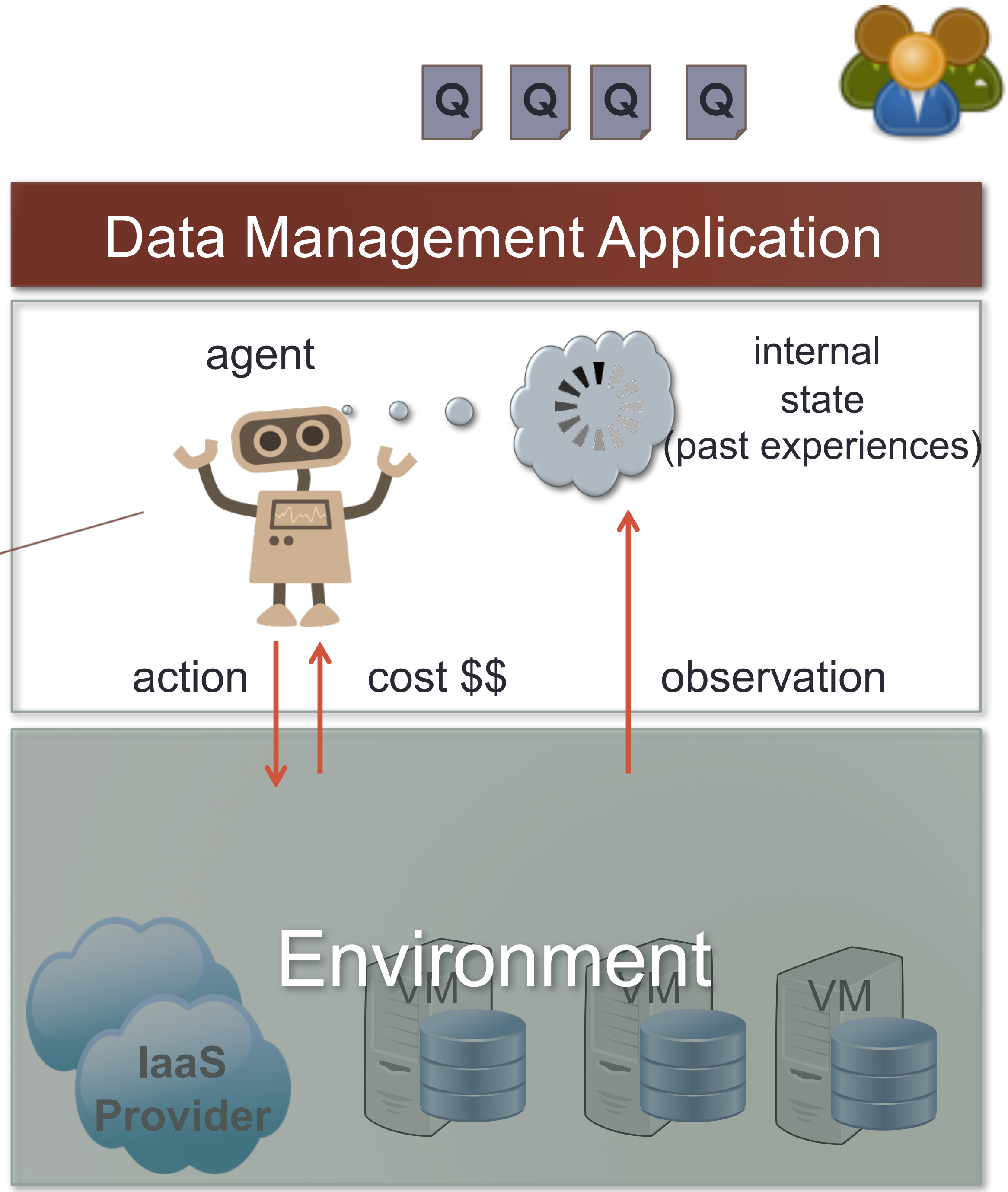
CMABs in WiSeDB

(Contextual Multi-Armed Bandits)

Contextual Multi-Armed Bandit Problem

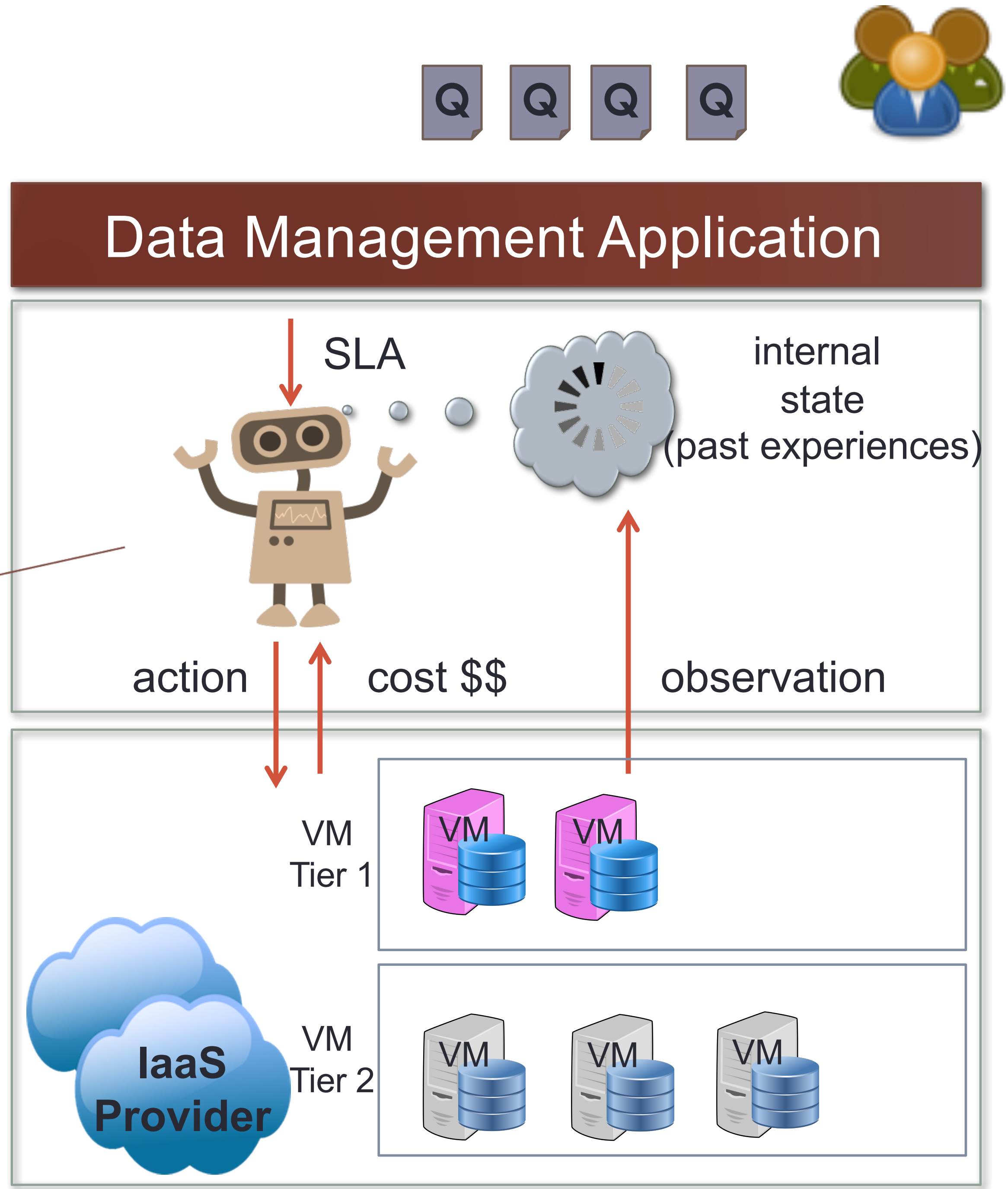
Slot Machine = Virtual Machine

*Which machine to use (new/ old) (**action**) so that you execute the incoming query with minimum cost $$ (**cost**)?*



CMABs in WiSeDB

(Contextual Multi-Armed Bandits)



Action (per VM)

- Accept
- Pass to next /new VM
- Down one VM tier

Reward

- $$ cost: processing & SLA violation penalties

Observation

- environment context (query, VM)
- action
- $$ cost

CMABs in WiSeDB

(Contextual Multi-Armed Bandits)

Action (per VM)

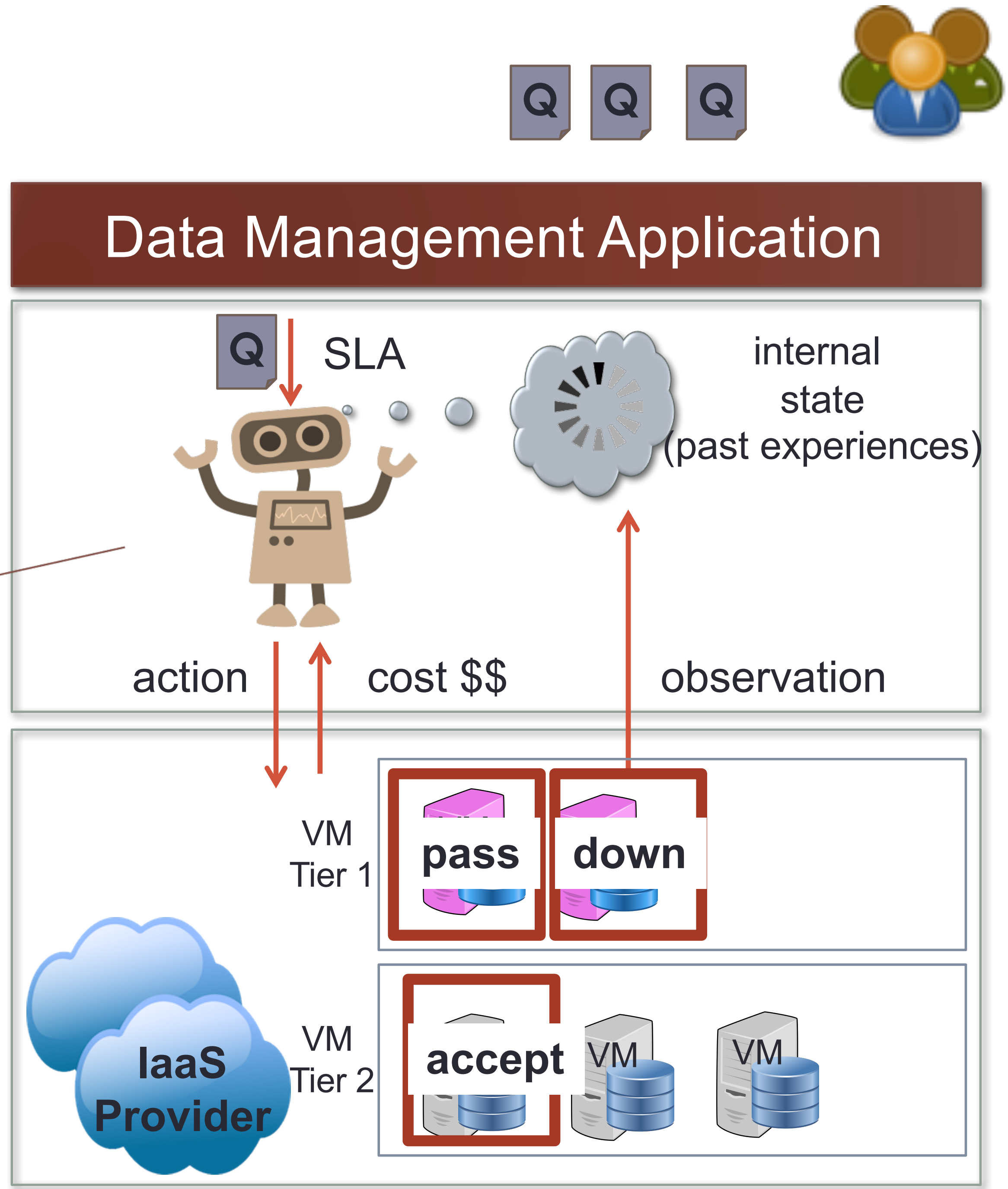
- Accept
- Pass to next /new VM
- Down one VM type

Reward

- $$ cost: processing & SLA violation penalties

Observation

- environment context (query, VM)
- action
- $$ cost

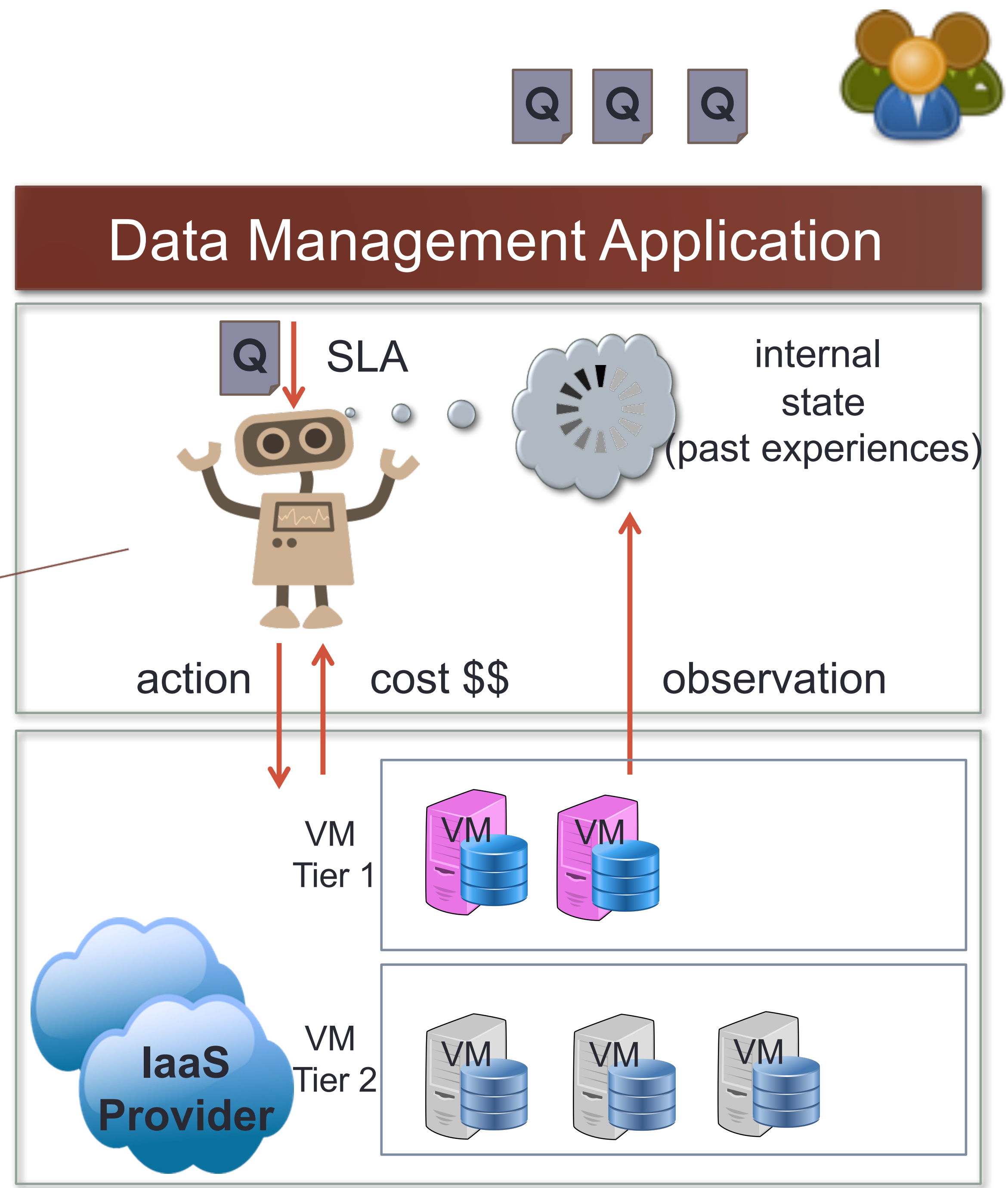


CMABs in WiSeDB
(Contextual Multi-Armed Bandits)

Action (per VM)
- Accept
- Pass to next /new VM
- Down one VM type

Reward
- $$ cost: processing & SLA violation penalties

Observation
- environment context (query, VM)
- action
- $$ cost

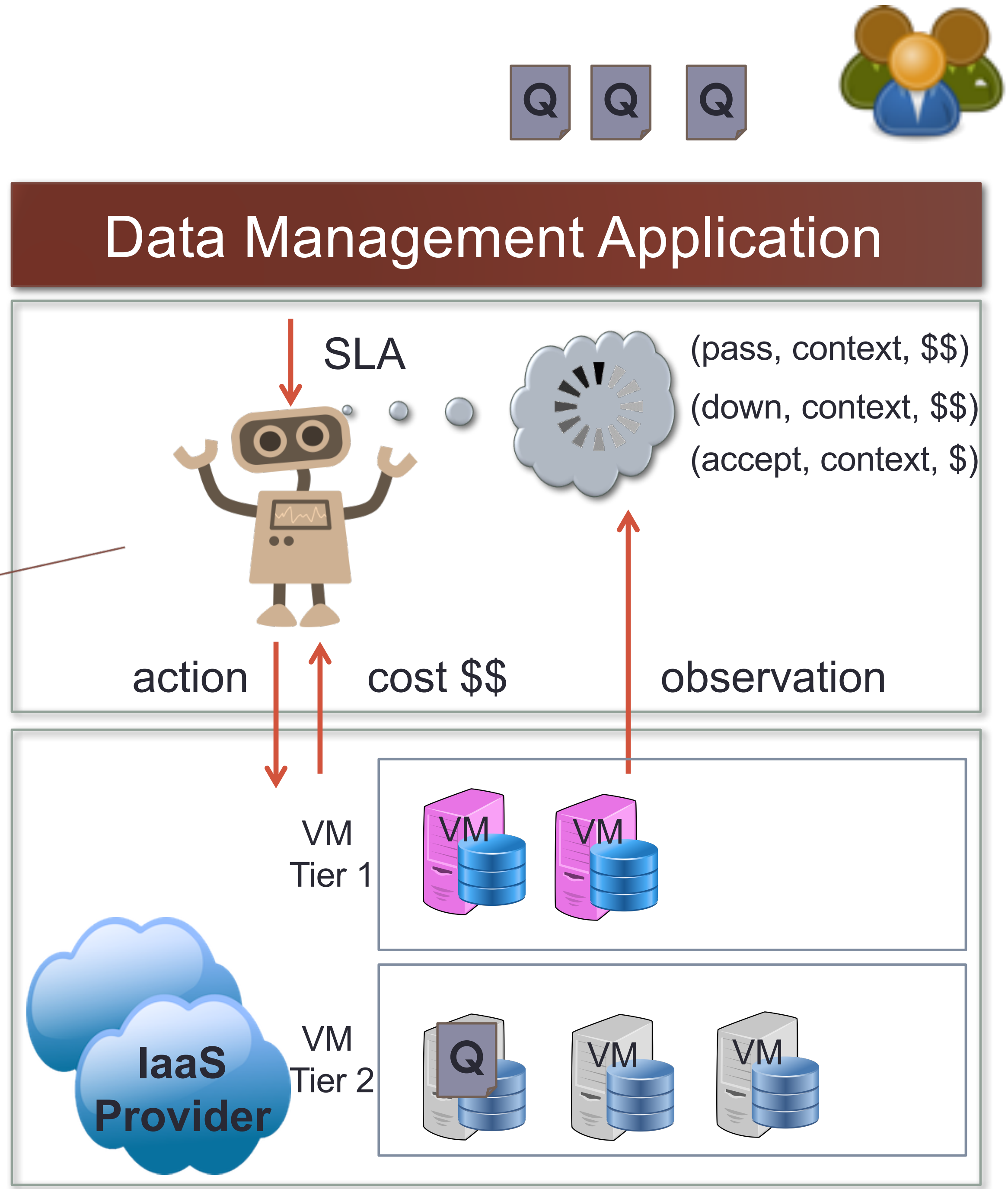


CMABs in WiSeDB
(Contextual Multi-Armed Bandits)

Action (per VM)
- Accept
- Pass to next /new VM
- Down one VM type

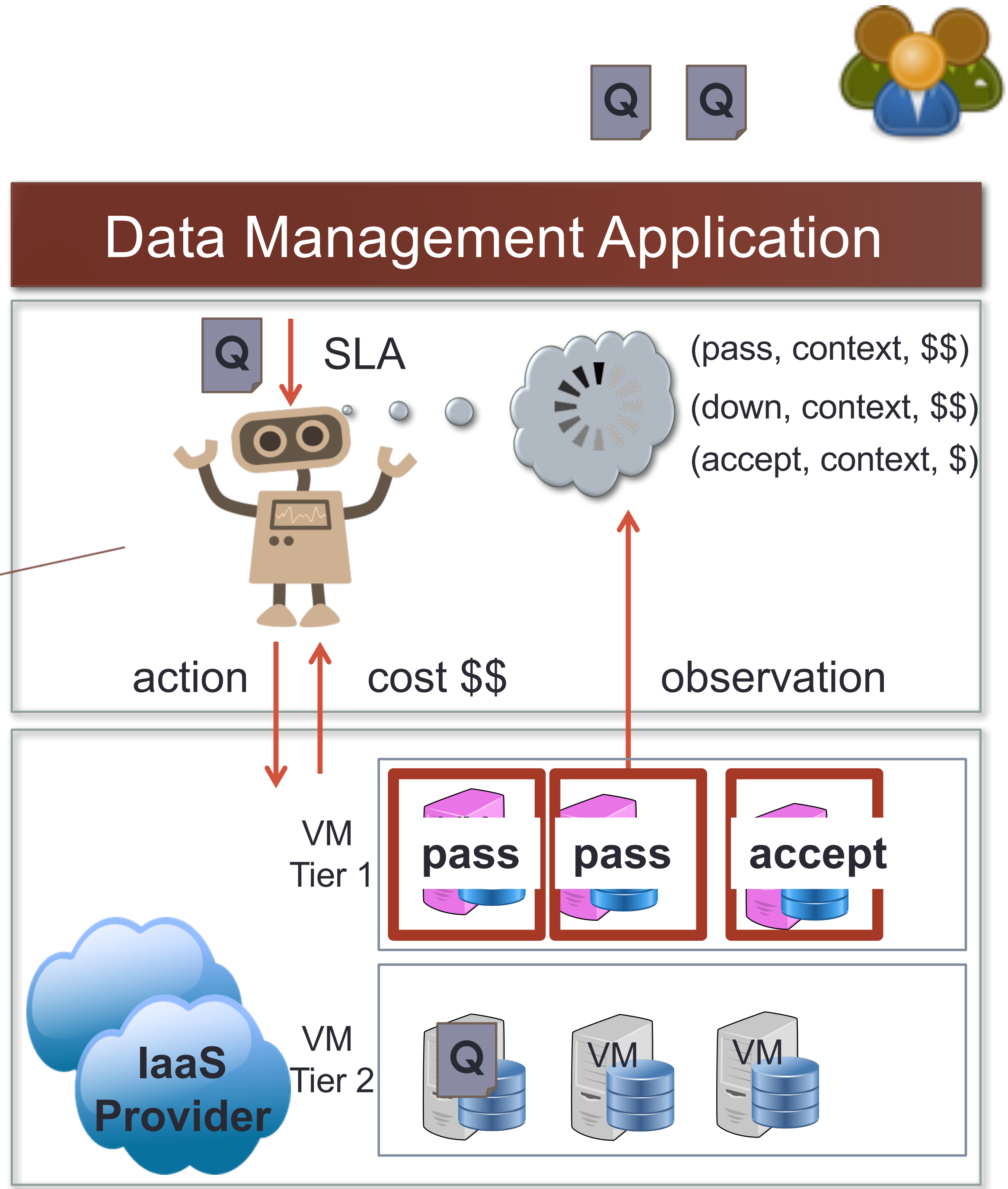
Reward
- $$ cost: processing & SLA violation penalties

Observation
- environment context (query, VM)
- action
- $$ cost



CMABs in WiSeDB

(Contextual Multi-Armed Bandits)



Action (per VM)

- Accept
- Pass to next /new VM
- Down one VM type

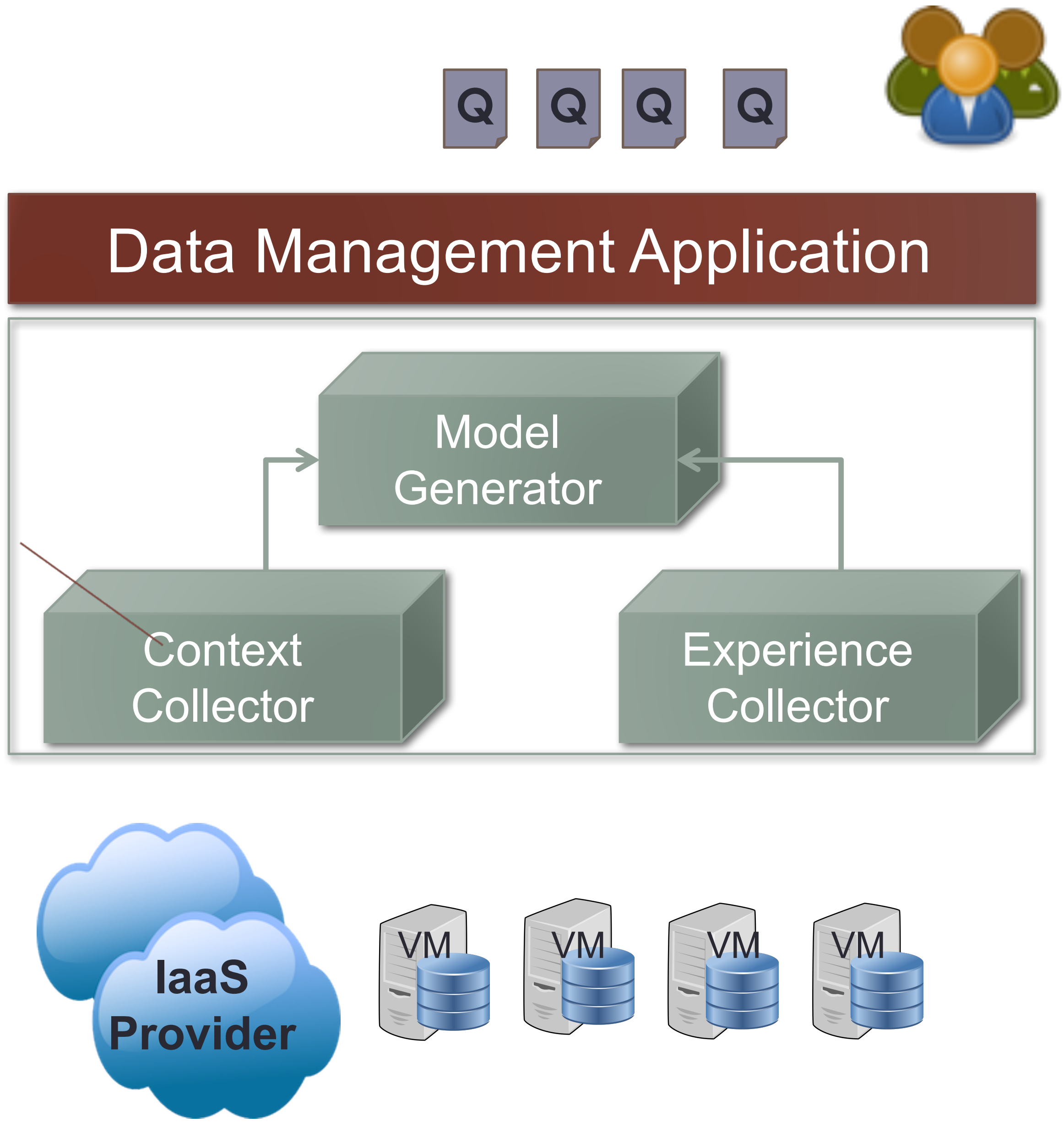
Reward

- $$ cost: processing & SLA violation penalties

Observation

- environment context (query, VM)
- action
- $$ cost

Online Learning



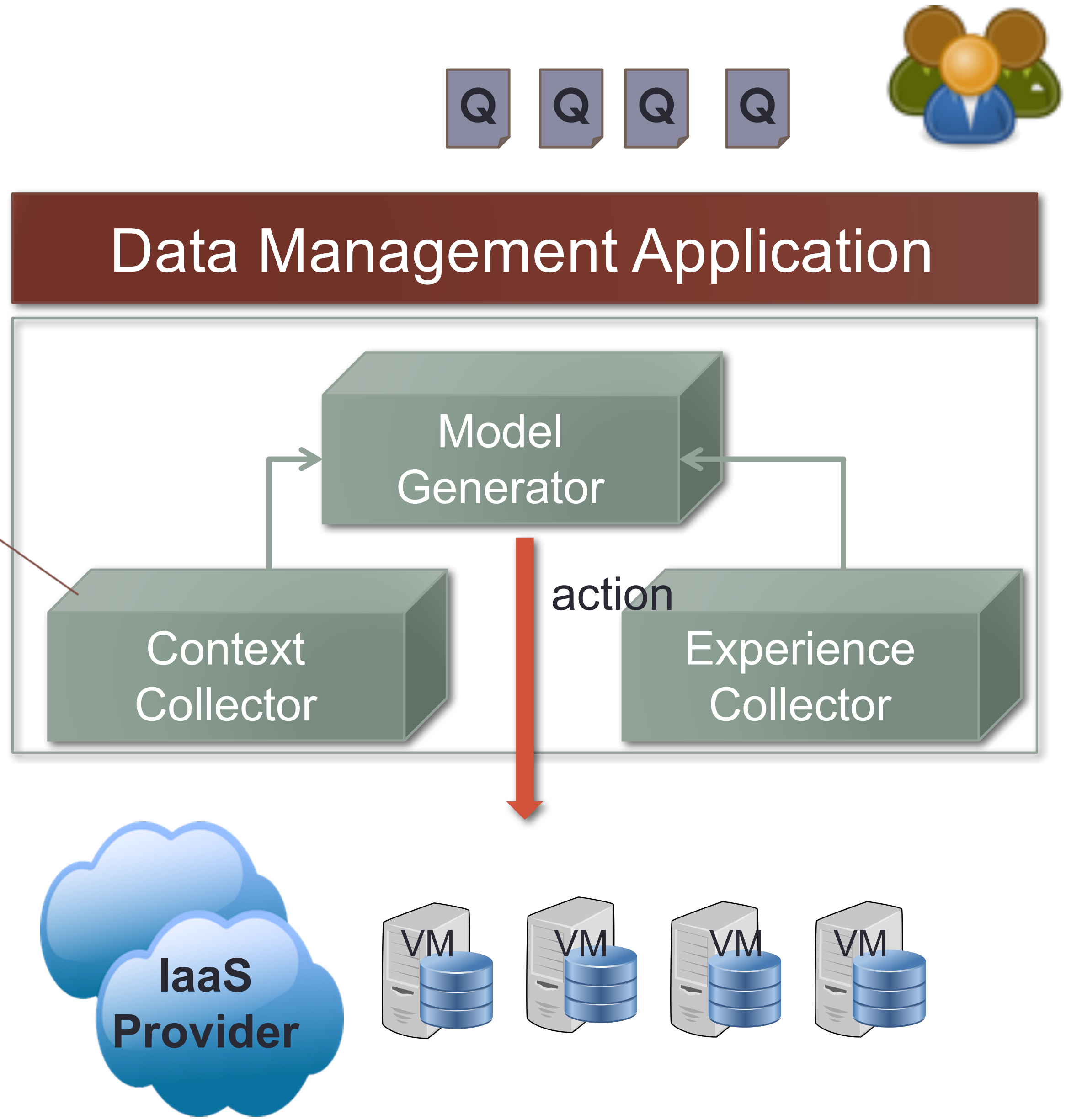
Context Features

- VM context
  - memory, I/O rate
  - #queries in queue
- Query context
  - tables used by current query
  - tables used by old query
  - # table scans
  - # joins
  - # spill joins
  - cache reads in the plan

Online Learning

Action Selection

- **Explore** opportunities
  - gather information
- **Exploit** “safe” actions
  - make best decision given current information



Probabilistic Action Selection

- Select action according to probability of being the best
- Past observations (action, context, cost) $D = \{(x_i, a_i, c_i)\}$
  - modeled by likelihood function over cost c : $P(c \mid \alpha, x, \theta)$
  - **θ: parameters of likelihood function: splits of a regression tree**
    - *if (# joins in the query =1) and (queries in the queue =3) => cost = $$*
- Posterior distribution of θ (Bayes rule)

$$P(\theta \mid D) \propto \prod P(c_i \mid a_i, x_i, \theta) P(\theta)$$

  - $P(\theta)$: prior distribution of parameters θ

perfect decision tree is unknown

- Choose action α' to minimize cost for perfect model θ*

$$\min_{a'} \mathrm{E}(c \mid a', x, \theta^*)]$$

Probabilistic Action Selection

- Exploitation: pick action based on mean of posterior $P(\theta|D)$

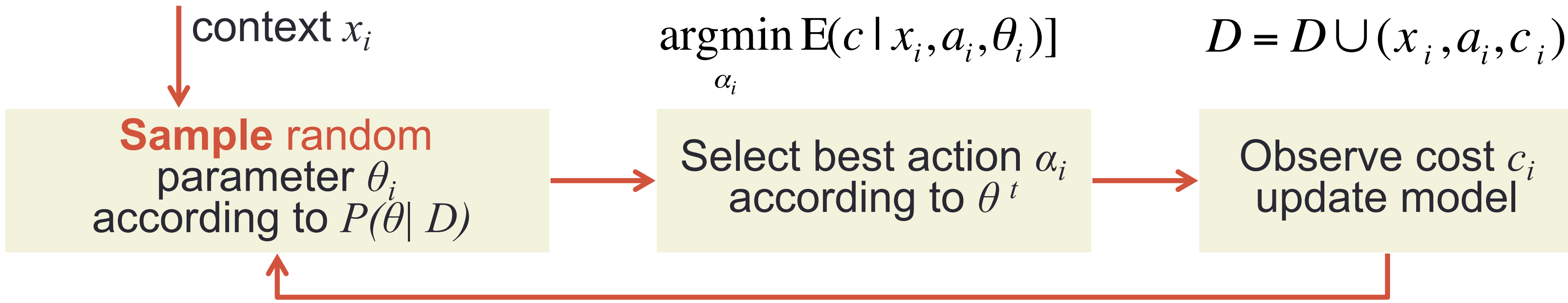
$$\min_{a'} \mathrm{E}(c \mid a', x) = \int \mathrm{E}(c \mid a', x, \theta) P(\theta \mid D) \, d\theta$$

- Exploration: pick a random action

- Thompson Sampling: balance exploration/exploitation

Select <u>random</u> action according to probability that it is the best

WiSeDB Action Selection



Select a random training set,
generate the regression tree and
pick best action according to it

Update the experience set

Create new model

Effectiveness

Training Data

30 query sequence

22 TPC-H templates

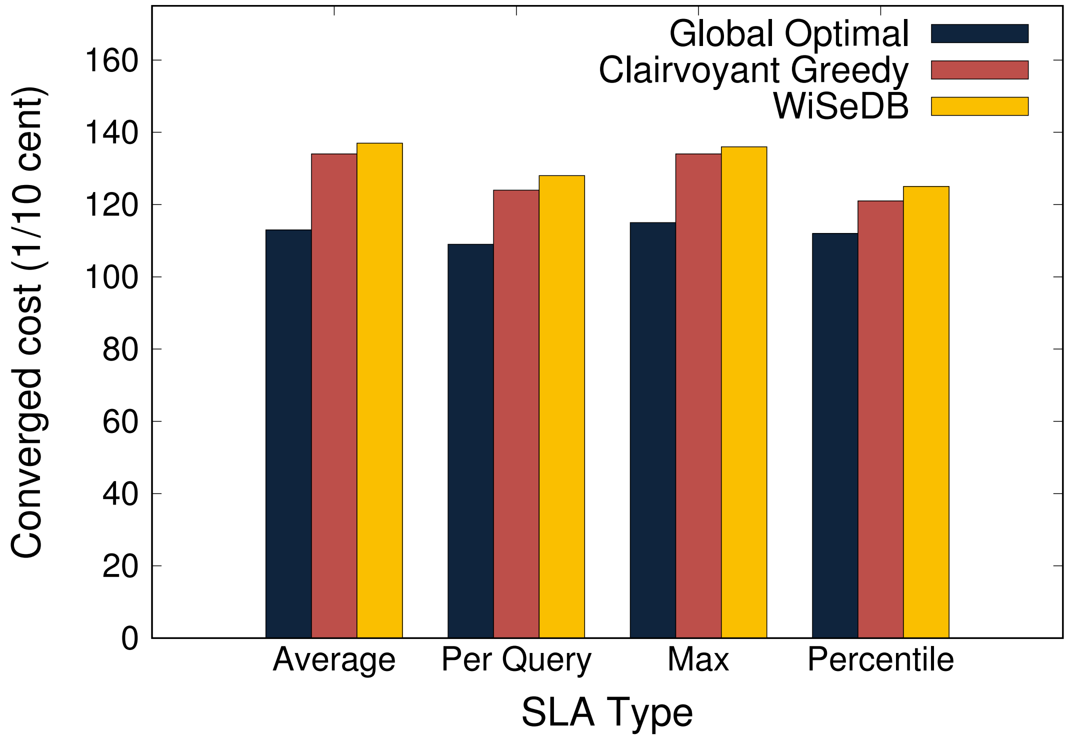
repeat until convergence

Optimal: **brute force (NP-hard)**

Clairvoyant: **perfect cost model**

Amazon AWS

t2.large, t2.medium, t2.small



WiSeDB models can perform at the same cost as a perfect cost model

Effectiveness (concurrency)

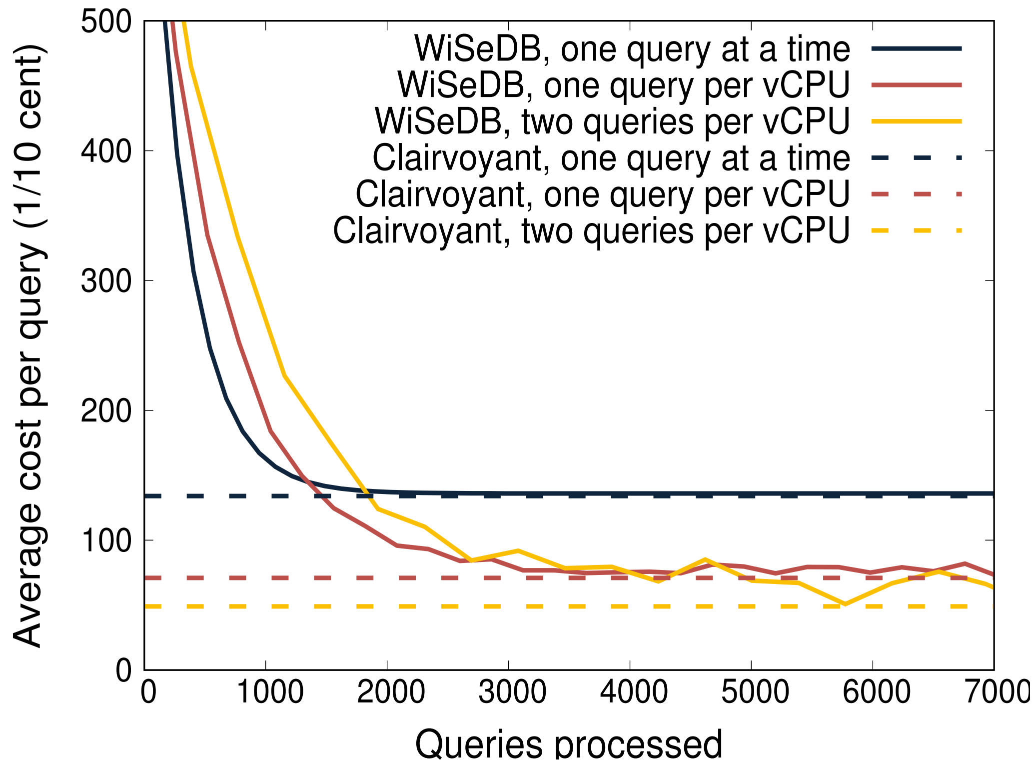
Training Data

22 TPC-H templates
900 queries/hour
Poison distribution

Clairvoyant: perfect cost model

One query/vCPU: 1-2 queries

Two queries/vCPU: 2-4 queries



WiSeDB models handles concurrency levels with no pre-training or tuning

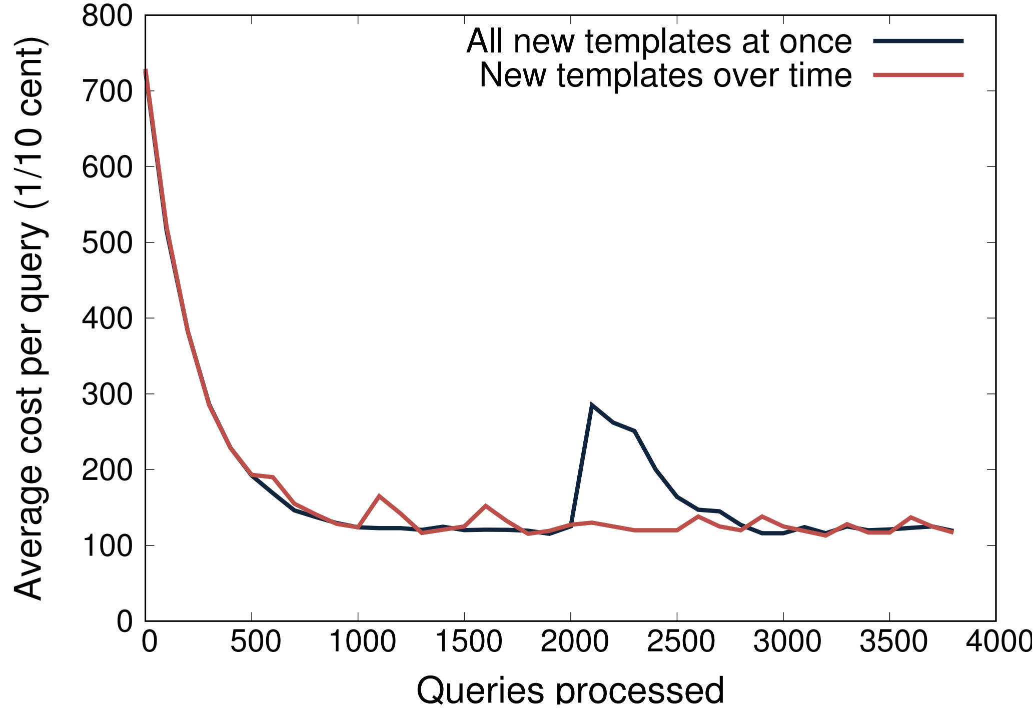
Adaptivity

<u>Training Data</u>

13 TPC-H templates
900 queries/hour
Poison distribution
Max SLO

all new at once: 7 new templates every 2000 queries (after convergence)

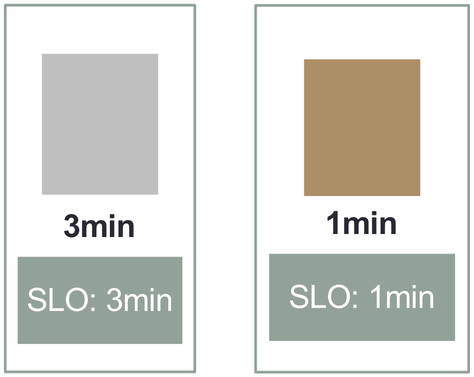
new over time: 1 new template every 500 queries

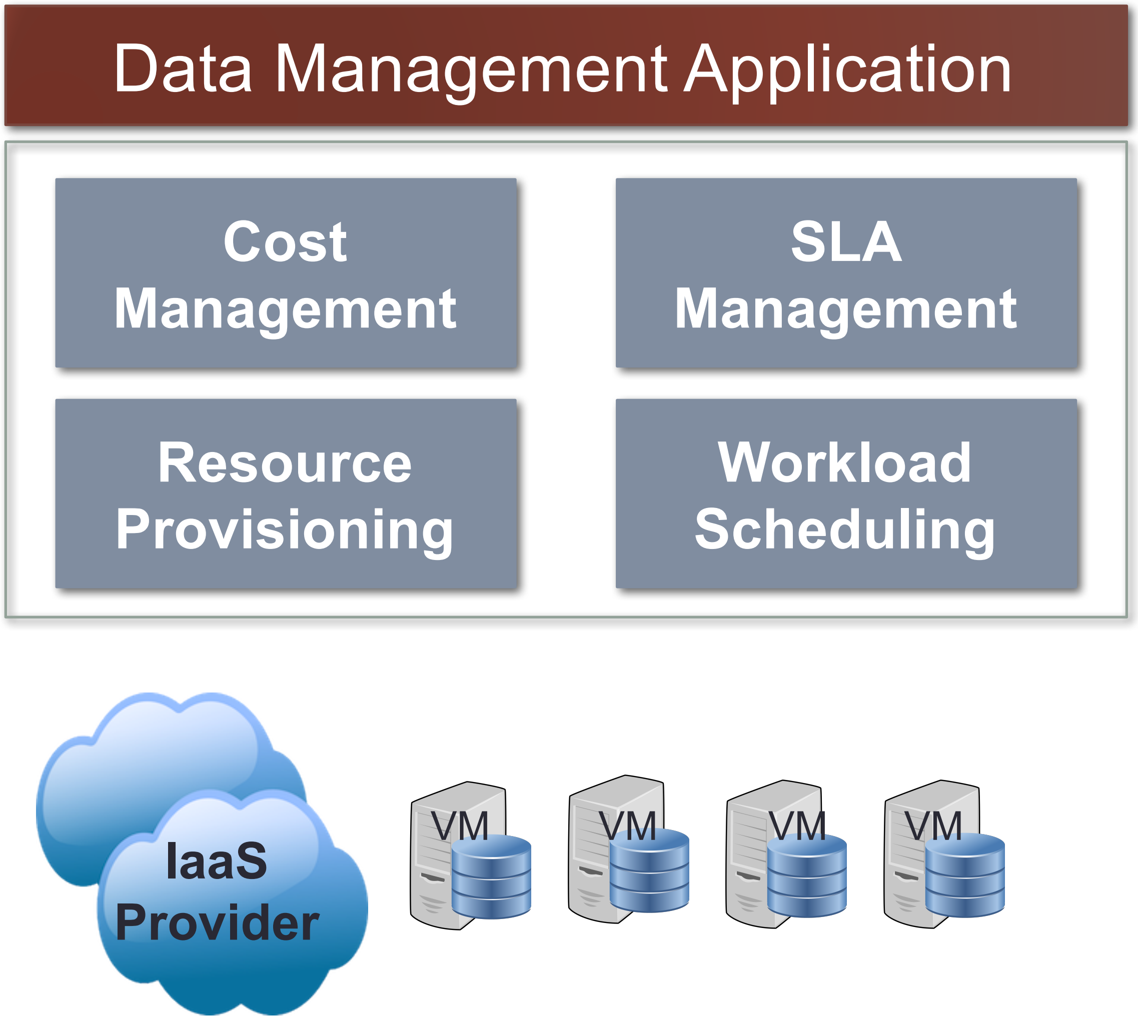


WiSeDB models quickly adapt to new unseen before templates

Next Steps: Prediction-free Batch Scheduling

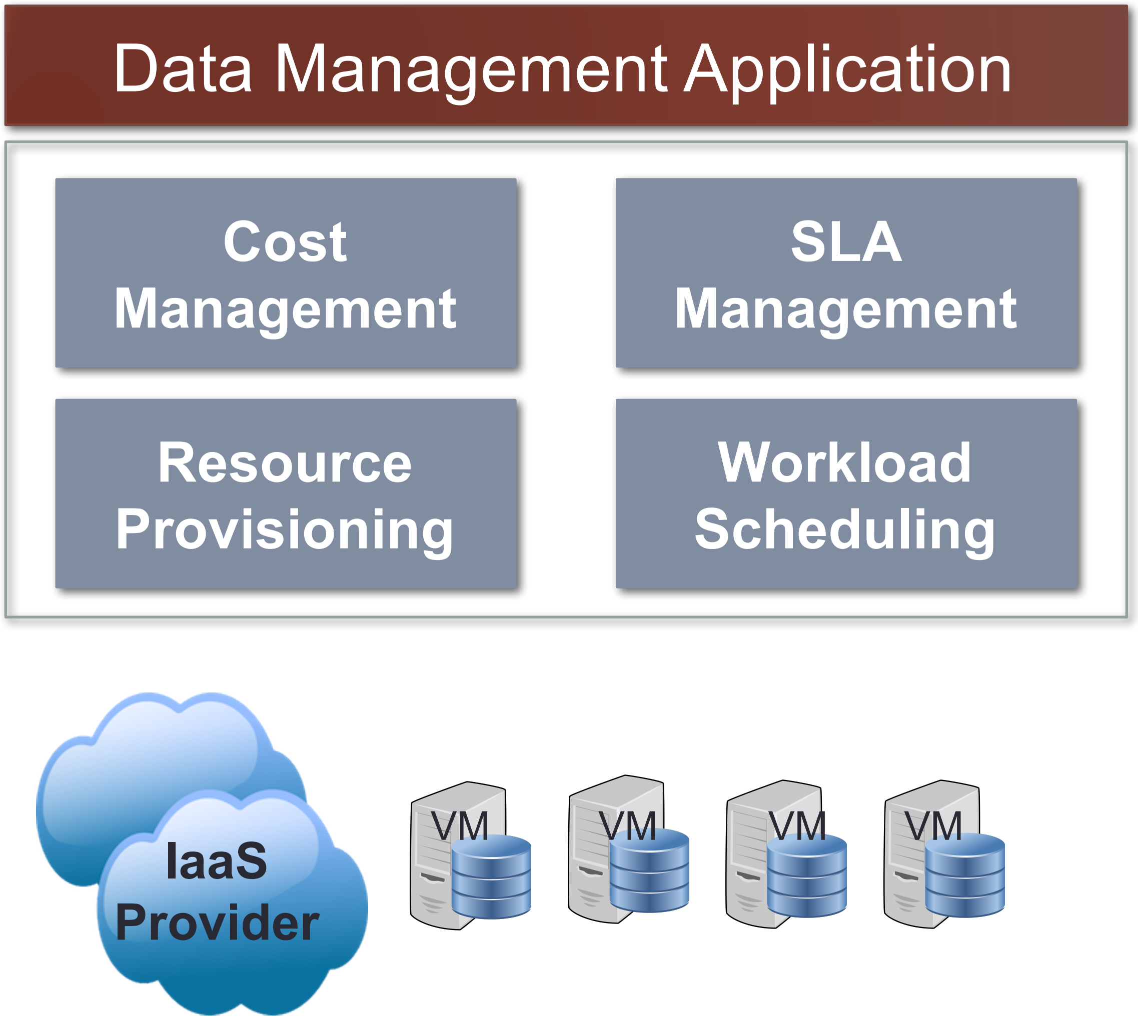
- Train once, use "**forever**"?
  - obsolescence detection and correction
- SVMs: Support Vector Machines
  - detect decision boundaries based on cost, SLO slack, SLA violation risk





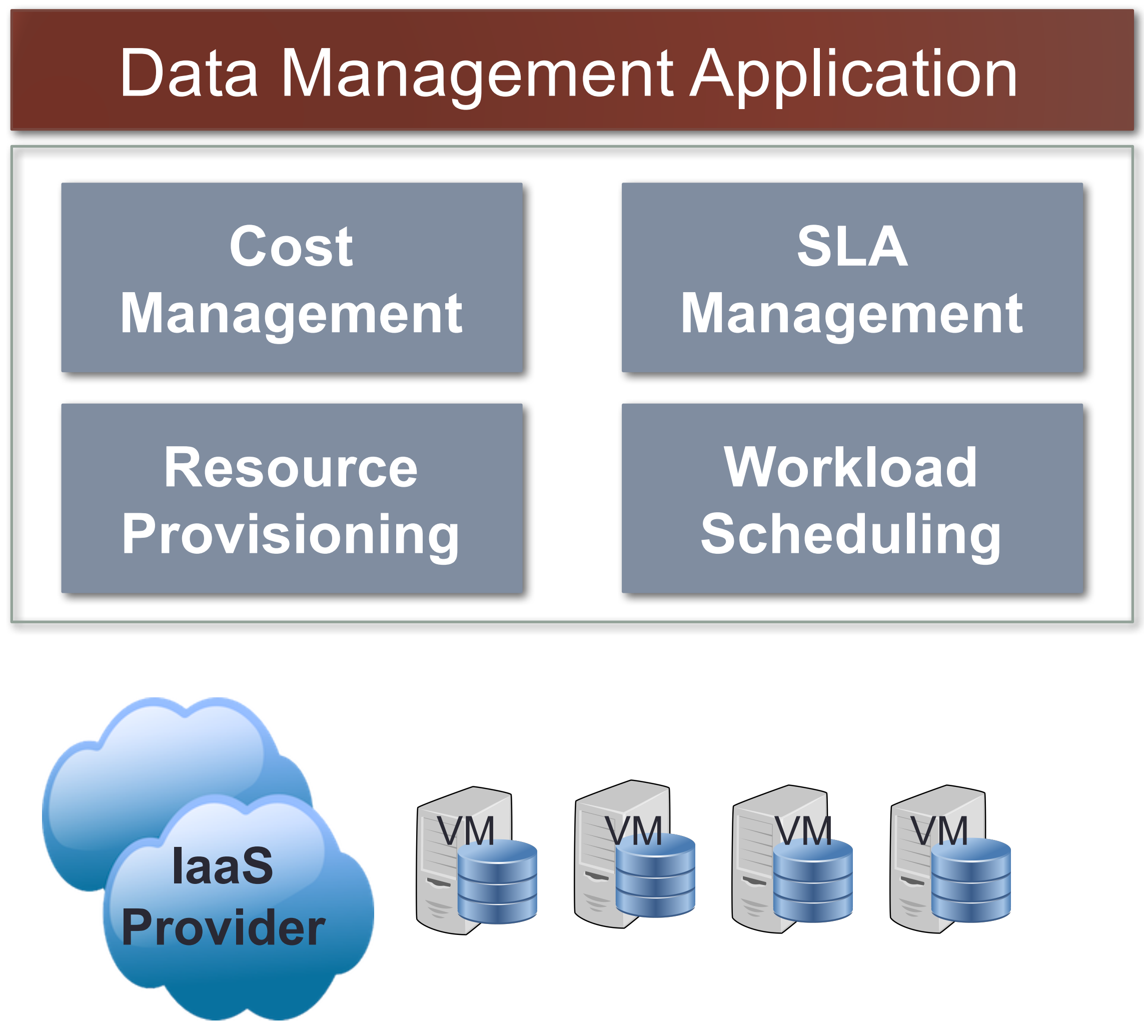
Next Steps: End-to-End Online Learning

- Query Scheduling
  - query ordering actions
- Shut-down strategy
  - hill-climbing learning
- Training overhead
  - search space reduction
  - warm bootstrapping



Next Steps: Learning-based Pricing

- Resource consumption & SLA pricing
- Predicted cost == minimum price
  - no SLA violation fees
- System & economics interplay
  - fairness & competition affects system design
  - “learn” the pricing scheme & system decisions that offers pricing fairness



Conclusions

- Cost and performance management for IaaS-deployed data managements apps are becoming more complex
  - human ability to derive insight remains the same

- WiSeDB demonstrates how **ML techniques can**
  - **offer insight** on the complex interplay of cost vs performance
  - **discover** customized solutions for app-specific SLAs
  - **automate** complex application management decisions
  - **adapt** to workload and resource configurations
  - **build** systems that perform beyond unaided human heuristics

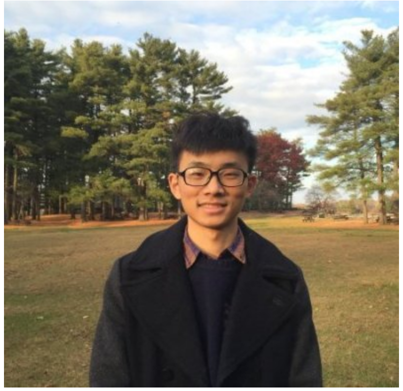
Our Database Group

Ryan Marcus

Cloud Databases
Machine Learning

Kyriaki Dimitriadou

Interactive Data Exploration
Machine Learning



Zhan Li

Benchmarking Optimizers
Statistical Analysis

THANK YOU

Questions?