WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON

CS 764: Topics in Database Management Systems
Lecture 24: Cloud OLTP

Xiangyao Yu

11/29/2021

Announcement

Mid-term grade announced

DAWN workshop schedule

- Online workshop using the lecture zoom link
- Reserve a presentation slot using the following google sheet
  - https://docs.google.com/spreadsheets/d/1BkO3ZqxNXxHRkI-XTnHmvQ1z66sS4LUVvIJiHS6HIJI/edit?usp=sharing
- Each group has a 10 min slot: **8 min presentation + 2 min Q/A**
- Live presentation preferred, but recording is also ok

Today's Paper: Cloud OLTP

An Evaluation of Alternative Architectures for Transaction Processing in the Cloud

Donald Kossmann Tim Kraska Simon Loesing

Systems Group, Department of Computer Science, ETH Zurich, Switzerland
{firstname.lastname}@inf.ethz.ch

ABSTRACT

Cloud computing promises a number of advantages for the deployment of data-intensive applications. One important promise is reduced cost with a pay-as-you-go business model. Another promise is (virtually) unlimited throughput by adding servers if the workload increases. This paper lists alternative architectures to effect cloud computing for database applications and reports on the results of a comprehensive evaluation of existing commercial cloud services that have adopted these architectures. The focus of this work is on transaction processing (i.e., read and update workloads), rather than analytics or OLAP workloads, which have recently gained a great deal of attention. The results are surprising in several ways. Most importantly, it seems that all major vendors have adopted a different architecture for their cloud services. As a result, the cost and performance of the services vary significantly depending on the workload.

Categories and Subject Descriptors

H.3.4 [**Systems and Software**]: Performance evaluation (efficiency and effectiveness); H.2.4 [**Systems**]: Distributed databases; K.6.0 [**General**]: Economics

General Terms

Experimentation, Measurement, Performance, Economics

Keywords

Cloud Computing, Benchmark, Performance Evaluation, Cloud Provider, Cloud DB, Transaction Processing, Cost

1. INTRODUCTION

Recently, there has been a great deal of hype about cloud computing. Cloud computing is on the top of Gartner's list of the ten most disruptive technologies of the next years [14]. All major software vendors and many start-ups have jumped on the bandwagon and claim that they are either cloud-enabled or cloud-enabling.

Cloud computing makes several promises. It promises a reduced time-to-market by removing or simplifying the time-consuming hardware provisioning, purchasing, and deployment processes. It promises cost reductions in several ways. First, it promises to turn capital costs into operational costs by adopting a pay-as-you-go business model. Second, it promises a better (close to 100 percent) utilization of the hardware resources. Cloud computing is, therefore, often considered a critical technology for green computing. Furthermore, cloud computing reduces operational cost and pain by automating IT tasks such as security patches and fail-over. In terms of performance, cloud computing promises (virtually) infinite scalability so that IT administrators need not worry about peak workloads. Finally, cloud computing promises improved flexibility in the utilization and management of both software and hardware which translates into savings in both time-to-market and cost.

As of today, a number of products have been launched. In particular, three of the big players of the IT industry, namely Amazon, Google, and Microsoft, have made product offerings. All these offerings have in common that they are available to a general audience by packaging cloud computing technology as a service, which can be activated from any personal computer via a simple REST interface. Also, all these offerings are geared towards delivering on the key promises of cloud computing and their adoption in the IT market place is rapidly growing.

The goal of this paper is to set a first yardstone in evaluating the current offerings. Using the database and workload of the TPC-W benchmark, we assessed Amazon, Google, and Microsoft's offerings and compared the results to the results obtained with a more traditional approach of running the TPC-W benchmark on a Java application server and an off-the-shelf relational database system. In particular, we wanted to address the following questions:

- How well do the offerings scale with an increasing workload? Can indeed a (virtually) infinite throughput be achieved?
- How expensive are these offerings and how does their cost / performance ratio (i.e., bang for the buck) compare?
- How predictable is the cost with regard to changes in the workload?

Obviously, the results reported in this paper are just a snapshot of the current state-of-the-art. The contribution is to establish a framework that allows vendors to gradually improve their services and allows users to compare products.

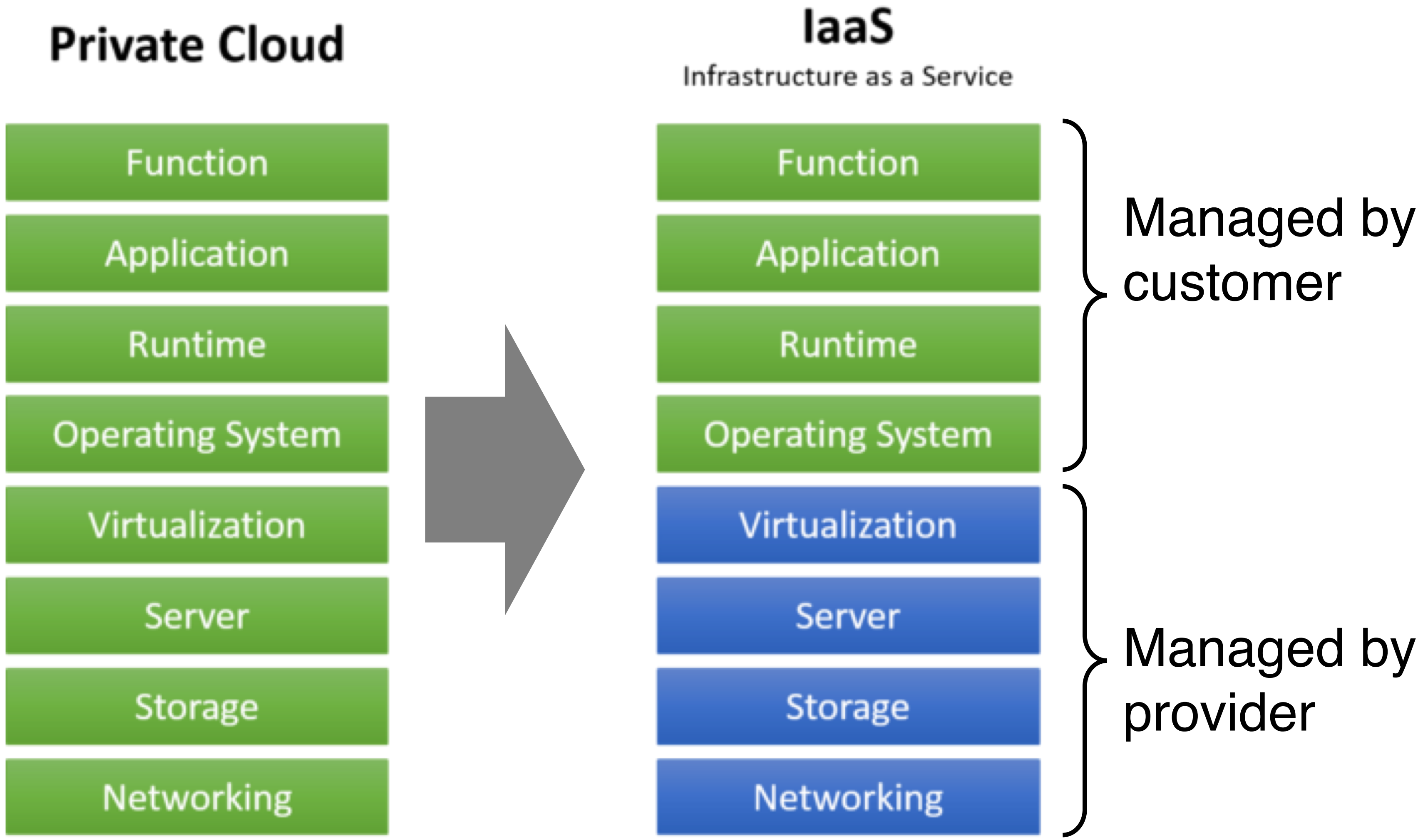
As will be shown, our experiments resulted in a number of surprises. Even though, many services look similar from the outside (e.g., Microsoft Azure and Amazon Web Services price matrixes are almost identical in terms of network bandwidth, storage cost, and CPU cost), the services vary dramatically when it comes to end-to-end performance, scalability, and cost. Maybe even more surprising are the differences in the architectures that effect large-scale data management and transaction workloads in the cloud.

SIGMOD 2010

Cloud Computing

Self-manage Hardware

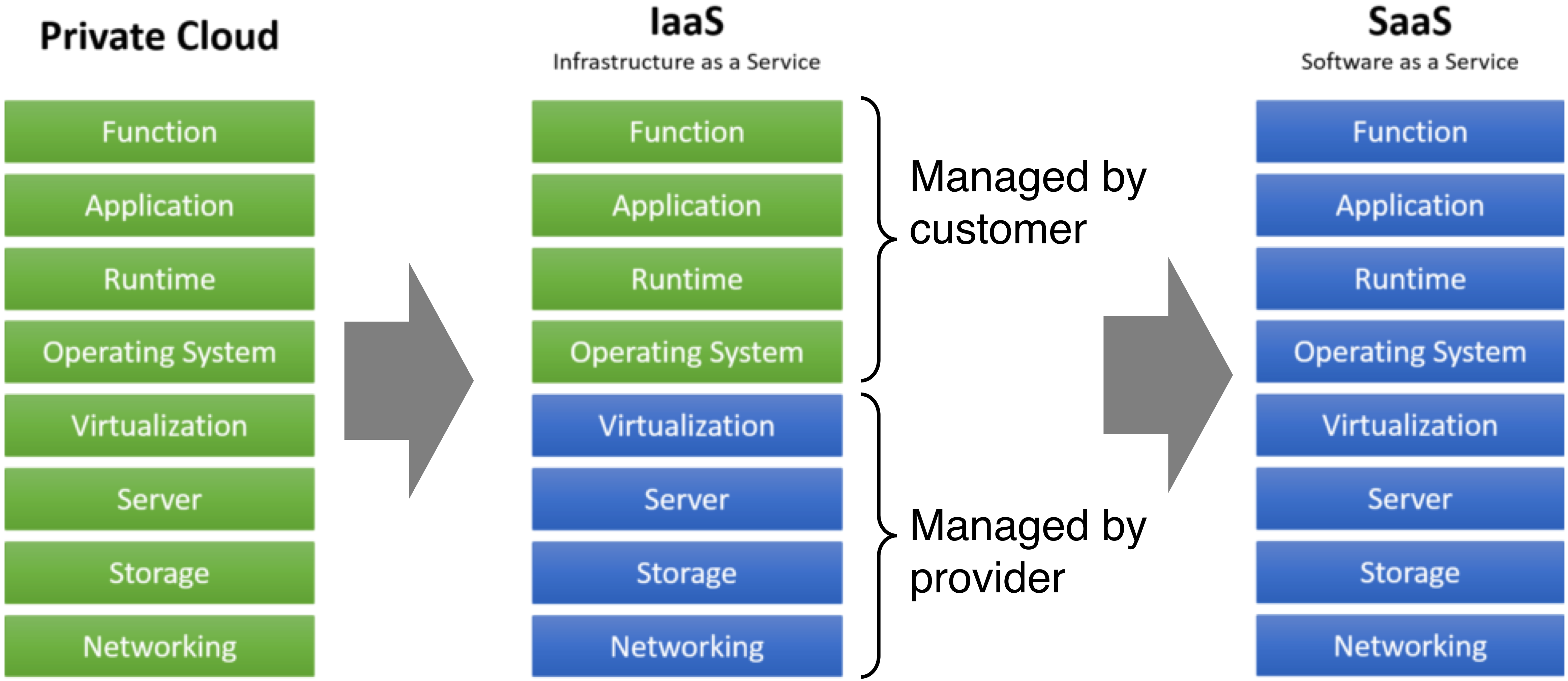
Cloud Computing



Self-manage Hardware

Self-deploy database

Cloud Computing



Self-manage Hardware | Self-deploy database | DB as a Service (DBaaS)

Databases Moving to the Cloud

Cloud databases can be **cheaper**

- Economies of scale
- Pay-as-you-go pricing model

Databases Moving to the Cloud

Cloud databases can be **cheaper**

- Economies of scale
- Pay-as-you-go pricing model

Cloud databases can be **more elastic**

- Unlimited storage and computation
- DB-as-a-Service simplifies administration and deployment

Databases Moving to the Cloud

Cloud databases can be **cheaper**

- Economies of scale
- Pay-as-you-go pricing model

Cloud databases can be **more elastic**

- Unlimited storage and computation
- DB-as-a-Service simplifies administration and deployment

Cloud databases can be **highly available**

- Geo-distribution with replication

Databases Moving to the Cloud

Cloud databases can be **cheaper**

- Economies of scale
- Pay-as-you-go pricing model

Cloud databases can be **more elastic**

- Unlimited storage and computation
- DB-as-a-Service simplifies administration and deployment

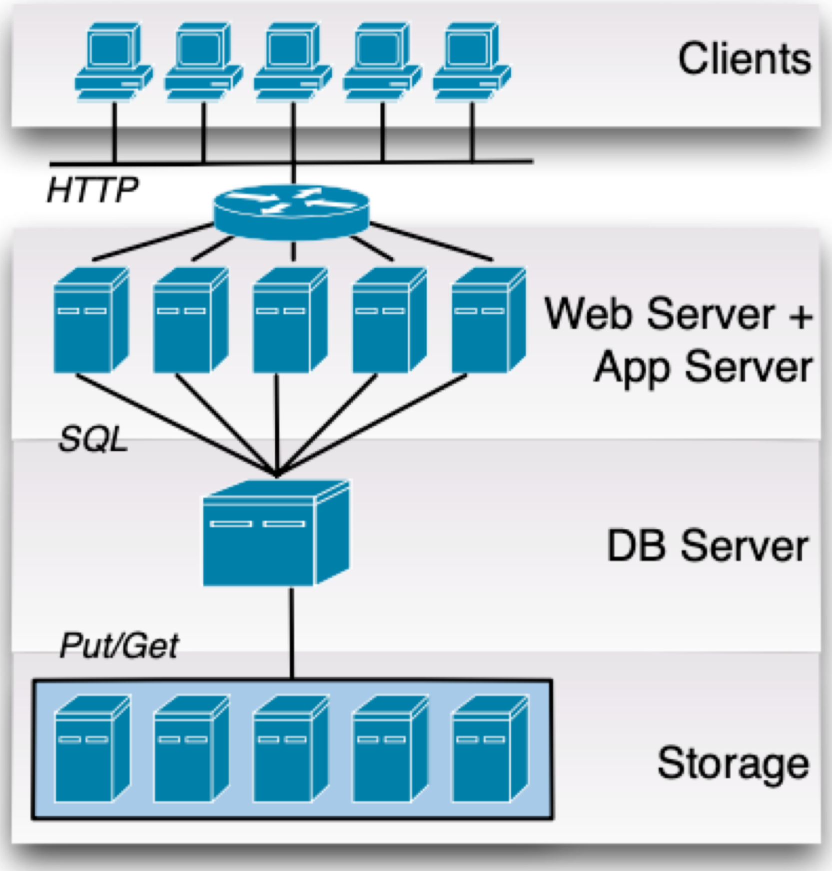
Cloud databases can be **highly available**

- Geo-distribution with replication

Cloud System Architectures

Layered architecture for cloud services

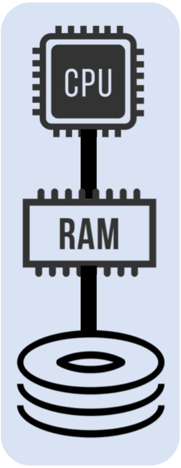
- Client layer
- Web/App server layer
- Database layer



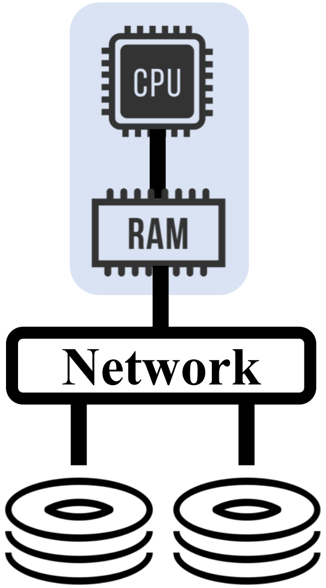
Database Architecture — Centralized

Centralized database

– Single-node database server
– Storage can be either **local** or through a **storage area network** (SAN)



Local storage

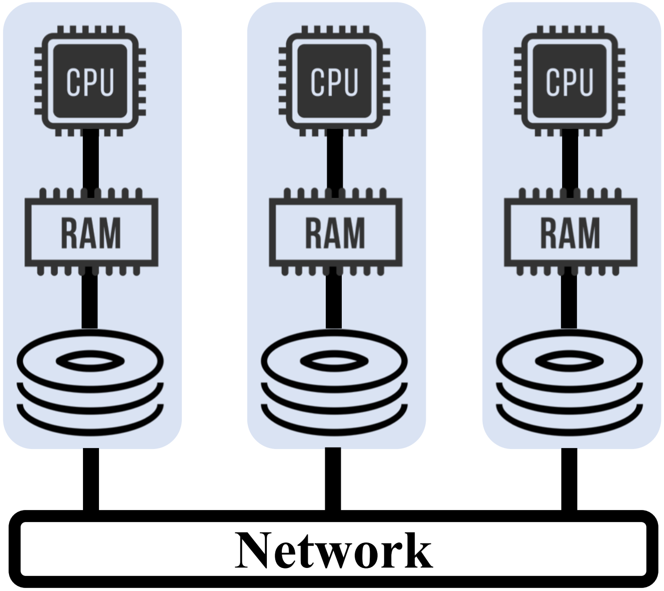


Storage Area Network (SAN)

Database Architecture — Partitioned

Partitioned database

- Shared-nothing architecture
- Data is partitioned across DB servers



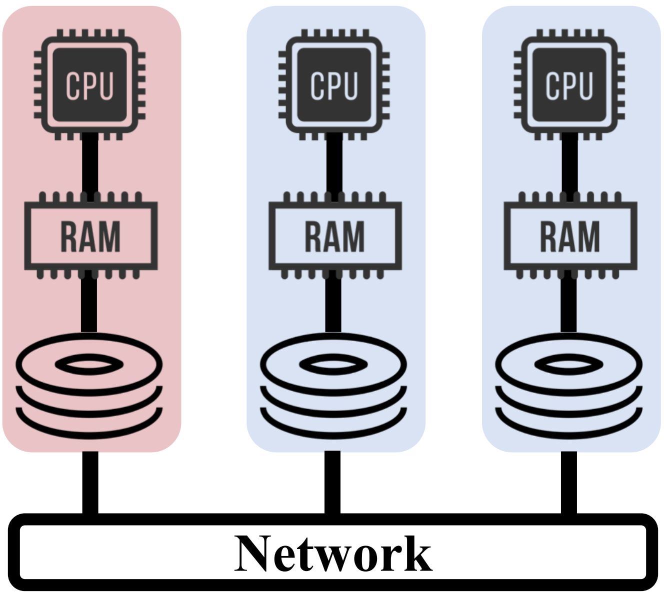
Shared Nothing

Database Architecture — Replication

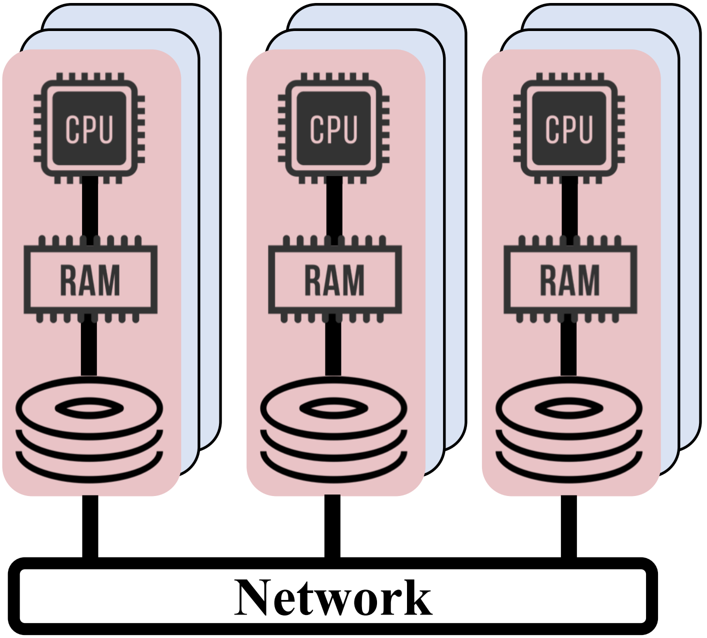
Database with replication

- Backup replicas can serve read-only requests
- Data is synchronously shipped to backup replicas

Primary Backup Backup



Single primary + replication

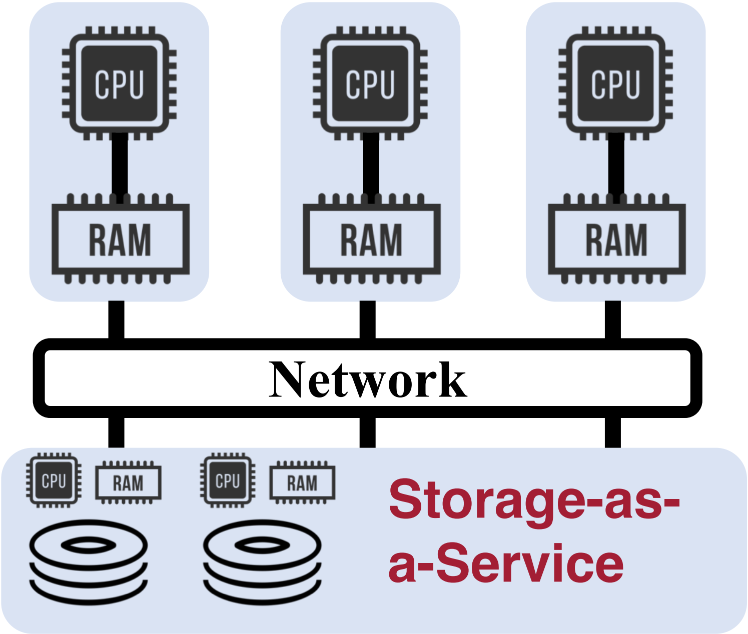


Partition + replication

Database Architecture — Storage Disaggregation

Storage disaggregation

- Storage and database compute are managed as two separate layers
- Computation and storage can scale independently



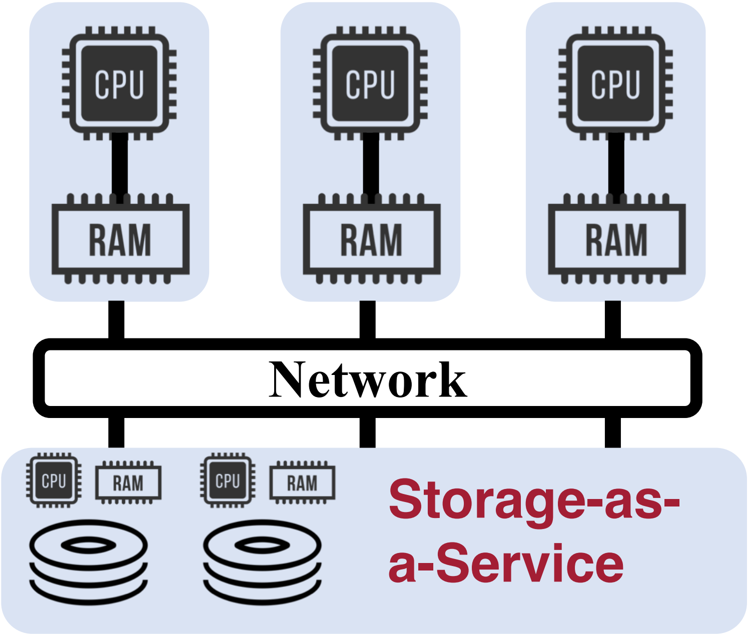
Database Architecture — Storage Disaggregation

Storage disaggregation

- Storage and database compute are managed as two separate layers
- Computation and storage can scale independently

Disaggregation vs. shared storage (e.g., SAN)?

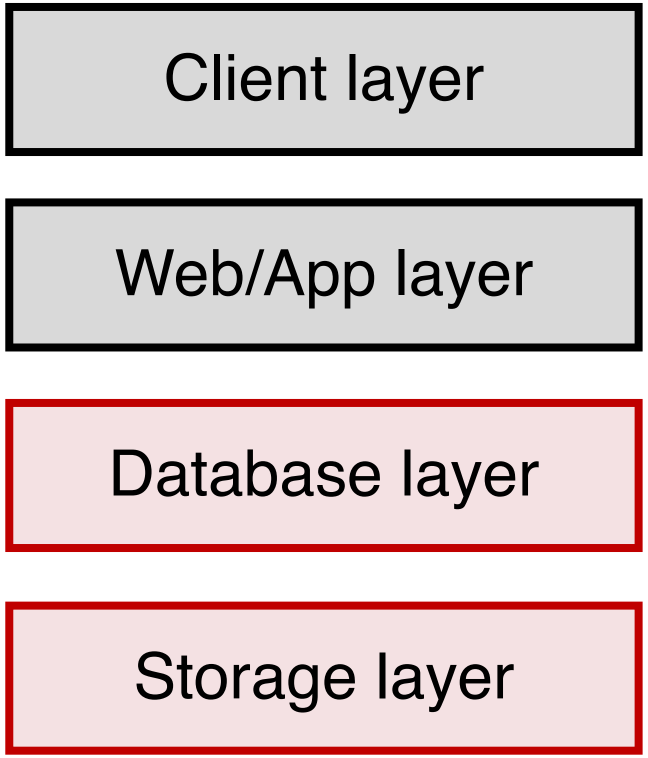
- Scaling of storage layer
- Built-in high availability
- Limited near-storage computation



A Simple Taxonomy

For database and storage layers

- Integrated or separate
- Single node or multiple nodes
- With or without replication



Revisit System Architecture

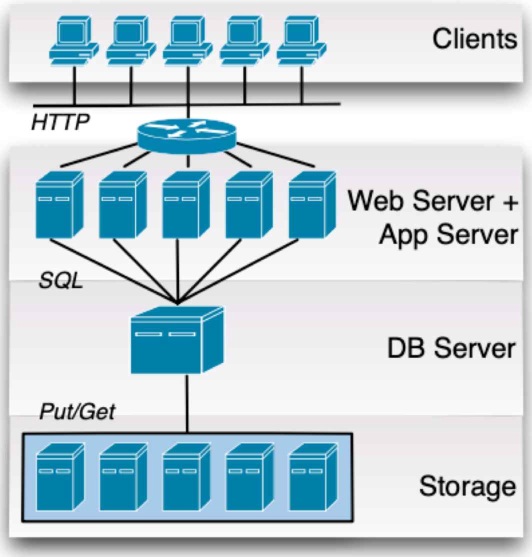


Figure 1: Classic Database Architecture

Simple taxonomy

- Integrated or separate
- Single node or multiple nodes
- With or without replication

Revisit System Architecture

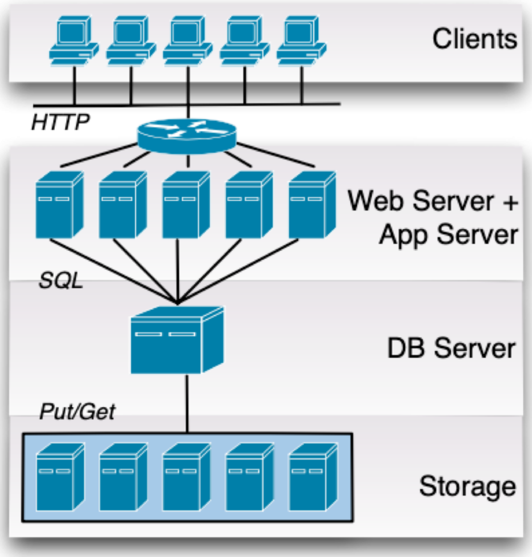


Figure 1: Classic Database Architecture

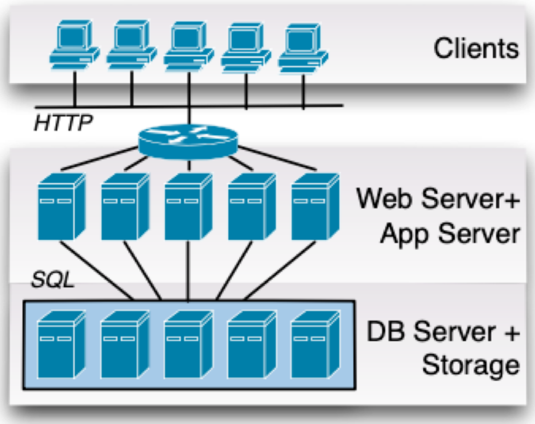


Figure 2: Partitioning

Simple taxonomy

- Integrated or separate
- Single node or multiple nodes
- With or without replication

Revisit System Architecture

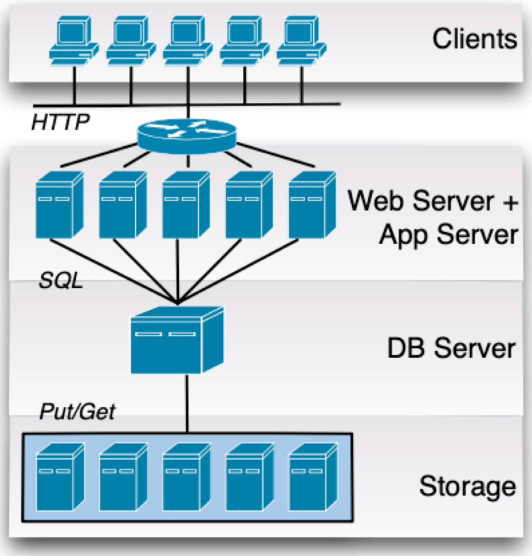


Figure 1: Classic Database Architecture

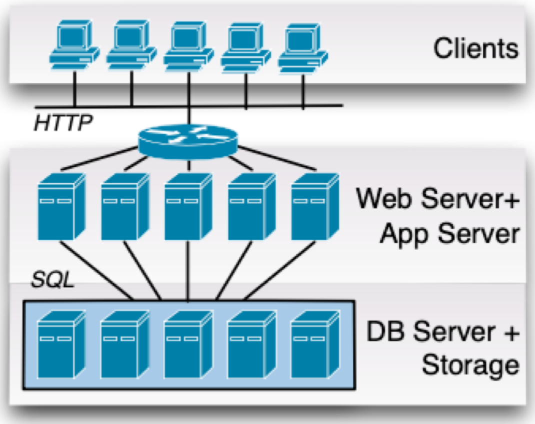


Figure 2: Partitioning

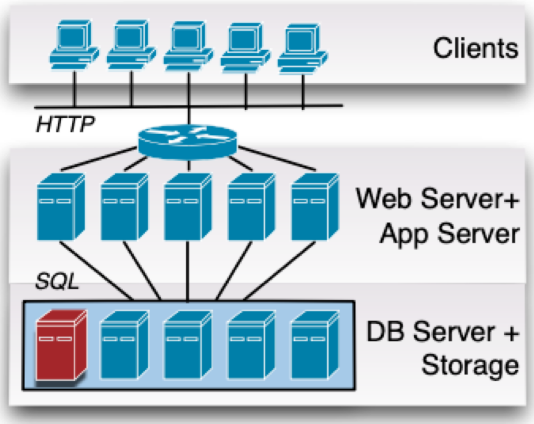


Figure 3: Replication

Simple taxonomy

- Integrated or separate
- Single node or multiple nodes
- With or without replication

Revisit System Architecture

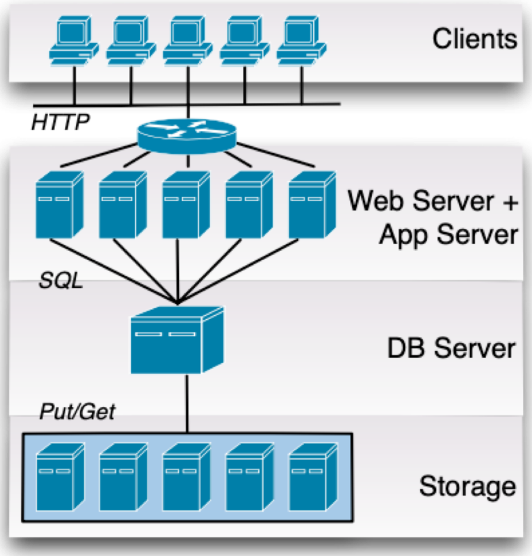


Figure 1: Classic Database Architecture

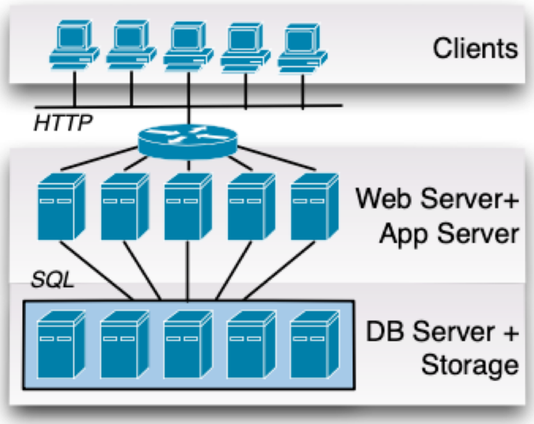


Figure 2: Partitioning

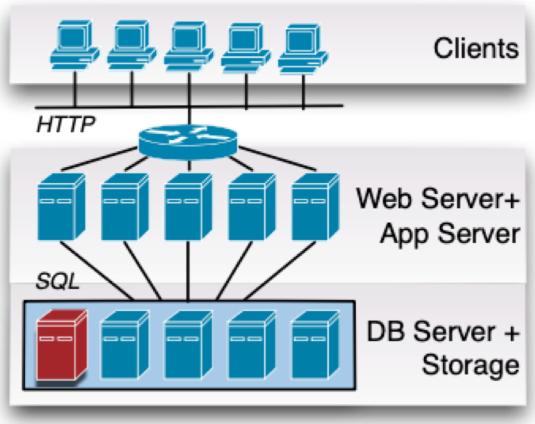


Figure 3: Replication

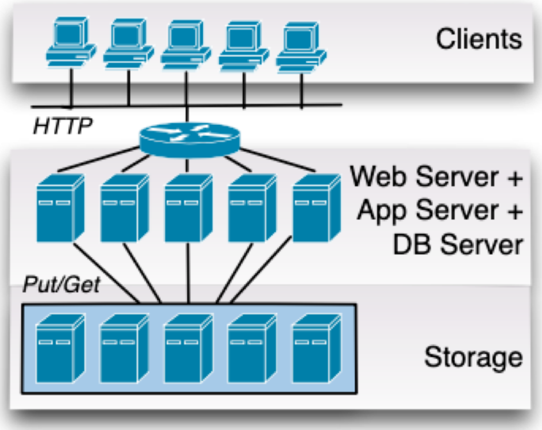


Figure 4: Distributed Control

Simple taxonomy

- Integrated or separate
- Single node or multiple nodes
- With or without replication

Revisit System Architecture

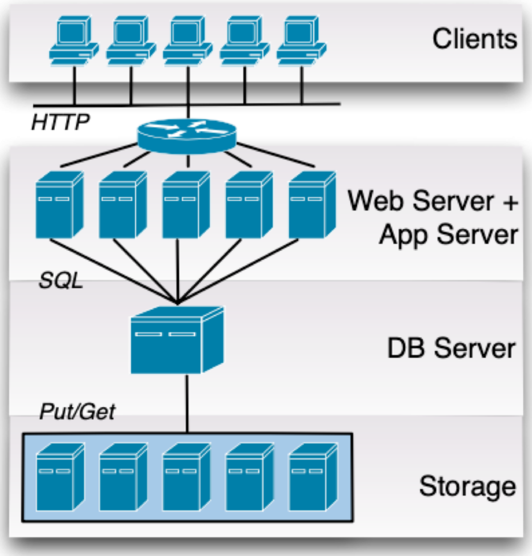


Figure 1: Classic Database Architecture

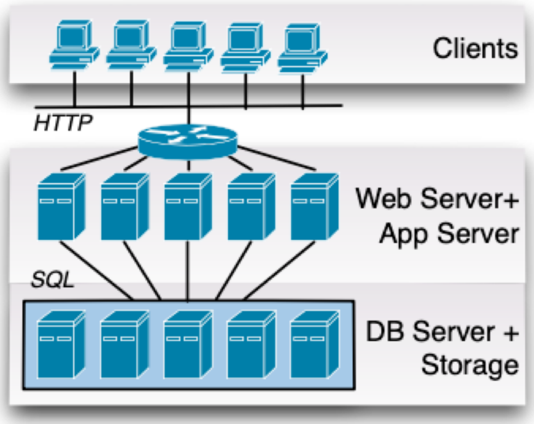


Figure 2: Partitioning

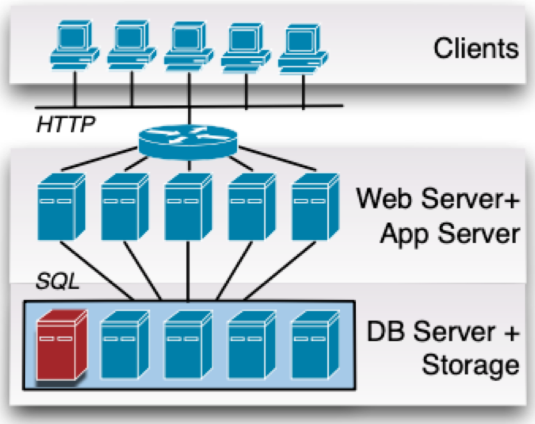


Figure 3: Replication

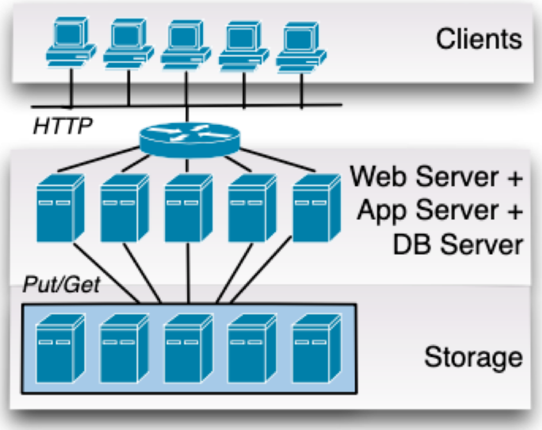


Figure 4: Distributed Control

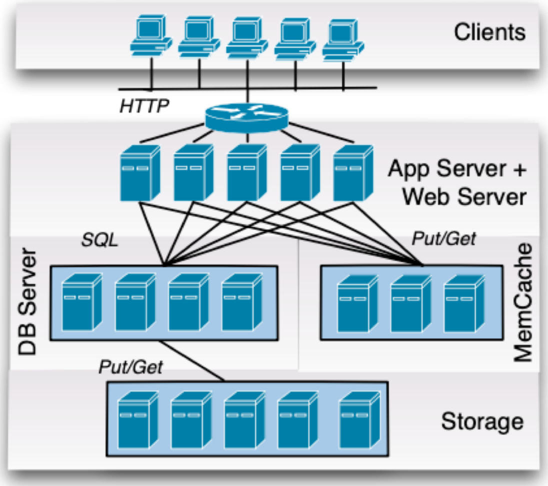


Figure 5: Caching

Simple taxonomy

- Integrated or separate
- Single node or multiple nodes
- With or without replication

Cloud Services

	AWS MySQL	AWS MySQL/R	AWS RDS	AWS SimpleDB	AWS S3	Google AppEng	MS Azure
Business Model	IaaS	IaaS	PaaS	PaaS	IaaS	PaaS	PaaS
Cloud Provider	Flexible	Flexible	Amazon	Amazon	Flexible	Google	Microsoft
Web/app server	Tomcat	Tomcat	Tomcat	Tomcat	Tomcat	AppEngine	.Net Azure
Database	MySQL	MySQL Rep	MySQL	SimpleDB	none	DataStore	SQL Azure
Storage / File Sys.	EBS	EC2 & EBS	-	-	S3	GFS	Windows Azure
Consistency	Repeatable Read	Repeatable Read	Repeatable Read	Eventual Consistency	Eventual Consistency	Snapshot Isolation	Snapshot Isolation
App-Language	Java	Java	Java	Java	Java	Java/AppEngine	C#
DB-Language	SQL	SQL	SQL	SimpleDB Queries	low-level API	GQL	SQL
Architecture	**Classic**	**Replication**	**Classic**	**Part.+Repl.**	**Distr. Contol**	**Part.+Repl.(+C)**	**Replication**
HW Config.	manual	manual	manual	manual/automatic	manual	automatic	manual/automatic

Table 1: Overview of Cloud Services

Cloud Services

	AWS MySQL	AWS MySQL/R	AWS RDS	AWS SimpleDB	AWS S3	Google AppEng	MS Azure
Business Model	IaaS	IaaS	PaaS	PaaS	IaaS	PaaS	PaaS
Cloud Provider	Flexible	Flexible	Amazon	Amazon	Flexible	Google	Microsoft
Web/app server	Tomcat	Tomcat	Tomcat	Tomcat	Tomcat	AppEngine	.Net Azure
Database	MySQL	MySQL Rep	MySQL	SimpleDB	none	DataStore	SQL Azure
Storage / File Sys.	EBS	EC2 & EBS	-	-	S3	GFS	Windows Azure
Consistency	Repeatable Read	Repeatable Read	Repeatable Read	Eventual Consistency	Eventual Consistency	Snapshot Isolation	Snapshot Isolation
App-Language	Java	Java	Java	Java	Java	Java/AppEngine	C#
DB-Language	SQL	SQL	SQL	SimpleDB Queries	low-level API	GQL	SQL
Architecture	**Classic**	**Replication**	**Classic**	**Part.+Repl.**	**Distr. Contol**	**Part.+Repl.(+C)**	**Replication**
HW Config.	manual	manual	manual	manual/automatic	manual	automatic	manual/automatic

Table 1: Overview of Cloud Services

Cloud Services

	AWS MySQL	AWS MySQL/R	AWS RDS	AWS SimpleDB	AWS S3	Google AppEng	MS Azure
Business Model	IaaS	IaaS	PaaS	PaaS	IaaS	PaaS	PaaS
Cloud Provider	Flexible	Flexible	Amazon	Amazon	Flexible	Google	Microsoft
Web/app server	Tomcat	Tomcat	Tomcat	Tomcat	Tomcat	AppEngine	.Net Azure
Database	MySQL	MySQL Rep	MySQL	SimpleDB	none	DataStore	SQL Azure
Storage / File Sys.	EBS	EC2 & EBS	-	-	S3	GFS	Windows Azure
Consistency	Repeatable Read	Repeatable Read	Repeatable Read	Eventual Consistency	Eventual Consistency	Snapshot Isolation	Snapshot Isolation
App-Language	Java	Java	Java	Java	Java	Java/AppEngine	C#
DB-Language	SQL	SQL	SQL	SimpleDB Queries	low-level API	GQL	SQL
Architecture	**Classic**	**Replication**	**Classic**	**Part.+Repl.**	**Distr. Contol**	**Part.+Repl.(+C)**	**Replication**
HW Config.	manual	manual	manual	manual/automatic	manual	automatic	manual/automatic

Table 1: Overview of Cloud Services

Cloud Services

	AWS MySQL	AWS MySQL/R	AWS RDS	AWS SimpleDB	AWS S3	Google AppEng	MS Azure
Business Model	IaaS	IaaS	PaaS	PaaS	IaaS	PaaS	PaaS
Cloud Provider	Flexible	Flexible	Amazon	Amazon	Flexible	Google	Microsoft
Web/app server	Tomcat	Tomcat	Tomcat	Tomcat	Tomcat	AppEngine	.Net Azure
Database	MySQL	MySQL Rep	MySQL	SimpleDB	none	DataStore	SQL Azure
Storage / File Sys.	EBS	EC2 & EBS	-	-	S3	GFS	Windows Azure
Consistency	Repeatable Read	Repeatable Read	Repeatable Read	Eventual Consistency	Eventual Consistency	Snapshot Isolation	Snapshot Isolation
App-Language	Java	Java	Java	Java	Java	Java/AppEngine	C#
DB-Language	SQL	SQL	SQL	SimpleDB Queries	low-level API	GQL	SQL
Architecture	**Classic**	**Replication**	**Classic**	**Part.+Repl.**	**Distr. Contol**	**Part.+Repl.(+C)**	**Replication**
HW Config.	manual	manual	manual	manual/automatic	manual	automatic	manual/automatic

Table 1: Overview of Cloud Services

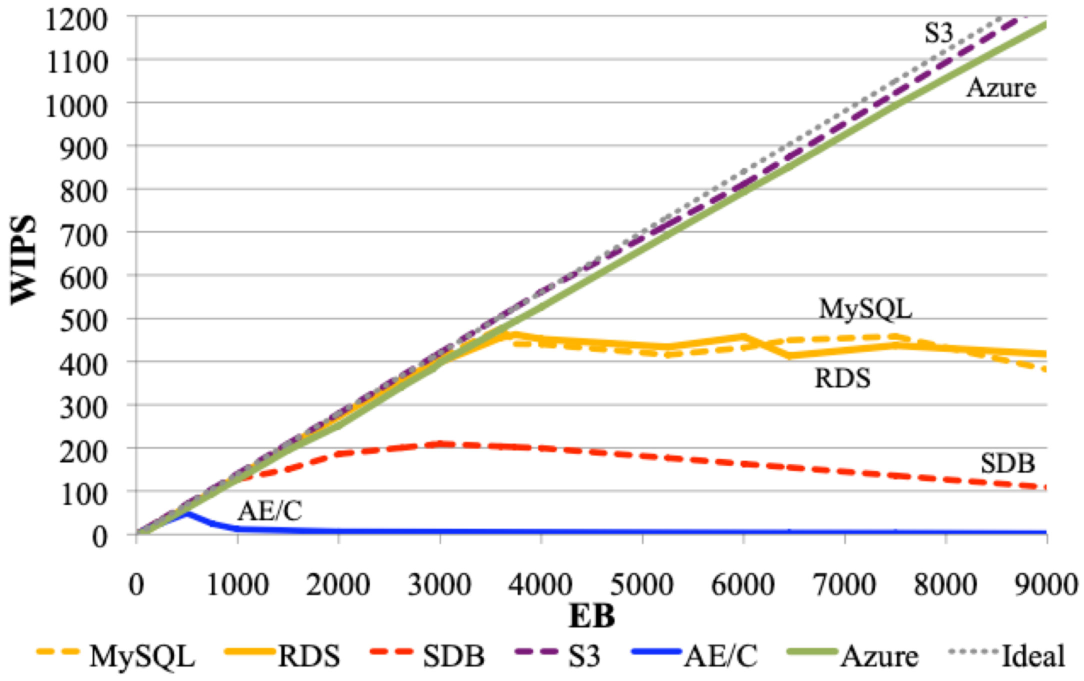
Evaluation

Designs that scale

- **S3**: storage disaggregation
- **Azure**: centralized but powerful primary node

Designs that do not scale

- **MySQL/RDS**: centralized primary is the bottleneck
- **SimpleDB/AppEngine**: excessive aborts



Cost Analysis

	1	10	100	500	1000
MySQL	0.635	0.072	***0.020***	***0.006***	***0.006***
MySQL/R	2.334	0.238	0.034	0.008	***0.006***
RDS	1.211	0.126	0.032	0.008	***0.006***
SimpleDB	0.384	0.073	0.042	0.039	0.037
S3	1.304	0.206	0.042	0.019	0.011
Google AE	0.002	0.028	0.033	0.042	0.176
Google AE/C	***0.002***	***0.018***	0.026	0.028	0.134
Azure	0.775	0.084	0.023	***0.006***	***0.006***

Table 3: Cost per WIPS [m$], Vary EB

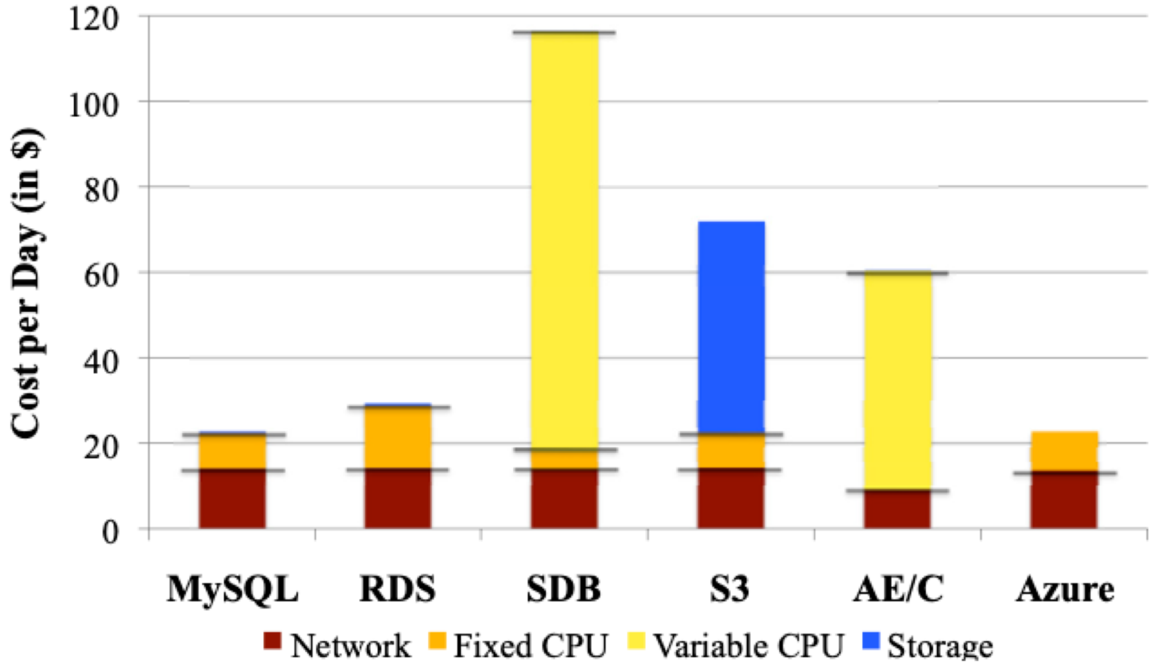


Figure 11: Cost Factors, 250 EBs [$]

Cost results are an artifact of the pricing model

Paper Conclusion

The more fundamental question of **what is the right data management architecture** for cloud computing could not be answered

Paper Conclusion

The more fundamental question of **what is the right data management architecture** for cloud computing could not be answered

	AWS MySQL	AWS MySQL/R	AWS RDS	AWS SimpleDB	AWS S3	Google AppEng	MS Azure
Business Model	IaaS	IaaS	PaaS	PaaS	IaaS	PaaS	PaaS
Cloud Provider	Flexible	Flexible	Amazon	Amazon	Flexible	Google	Microsoft
Web/app server	Tomcat	Tomcat	Tomcat	Tomcat	Tomcat	AppEngine	.Net Azure
Database	MySQL	MySQL Rep	MySQL	SimpleDB	none	DataStore	SQL Azure
Storage / File Sys.	EBS	EC2 & EBS	-	-	S3	GFS	Windows Azure
Consistency	Repeatable Read	Repeatable Read	Repeatable Read	Eventual Consistency	Eventual Consistency	Snapshot Isolation	Snapshot Isolation
App-Language	Java	Java	Java	Java	Java	Java/AppEngine	C#
DB-Language	SQL	SQL	SQL	SimpleDB Queries	low-level API	GQL	SQL
Architecture	**Classic**	**Replication**	**Classic**	**Part.+Repl.**	**Distr. Contol**	**Part.+Repl.(+C)**	**Replication**
HW Config.	manual	manual	manual	manual/automatic	manual	automatic	manual/automatic

Table 1: Overview of Cloud Services

Single primary bottleneck

Paper Conclusion

The more fundamental question of **what is the right data management architecture** for cloud computing could not be answered

	AWS MySQL	AWS MySQL/R	AWS RDS	AWS SimpleDB	AWS S3	Google AppEng	MS Azure
Business Model	IaaS	IaaS	PaaS	PaaS	IaaS	PaaS	PaaS
Cloud Provider	Flexible	Flexible	Amazon	Amazon	Flexible	Google	Microsoft
Web/app server	Tomcat	Tomcat	Tomcat	Tomcat	Tomcat	AppEngine	.Net Azure
Database	MySQL	MySQL Rep	MySQL	SimpleDB	none	DataStore	SQL Azure
Storage / File Sys.	EBS	EC2 & EBS	-	-	S3	GFS	Windows Azure
Consistency	Repeatable Read	Repeatable Read	Repeatable Read	Eventual Consistency	Eventual Consistency	Snapshot Isolation	Snapshot Isolation
App-Language	Java	Java	Java	Java	Java	Java/AppEngine	C#
DB-Language	SQL	SQL	SQL	SimpleDB Queries	low-level API	GQL	SQL
Architecture	**Classic**	**Replication**	**Classic**	**Part.+Repl.**	**Distr. Contol**	**Part.+Repl.(+C)**	**Replication**
HW Config.	manual	manual	manual	manual/automatic	manual	automatic	manual/automatic

Table 1: Overview of Cloud Services

Single primary bottleneck

Excessive aborts

Paper Conclusion

The more fundamental question of **what is the right data management architecture** for cloud computing could not be answered

No strong consistency

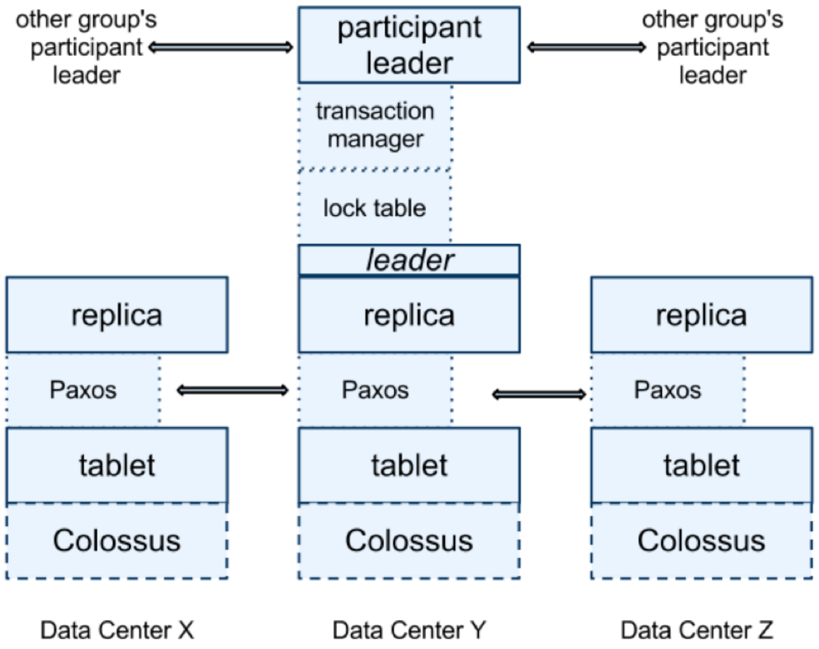
	AWS MySQL	AWS MySQL/R	AWS RDS	AWS SimpleDB	AWS S3	Google AppEng	MS Azure
Business Model	IaaS	IaaS	PaaS	PaaS	IaaS	PaaS	PaaS
Cloud Provider	Flexible	Flexible	Amazon	Amazon	Flexible	Google	Microsoft
Web/app server	Tomcat	Tomcat	Tomcat	Tomcat	Tomcat	AppEngine	.Net Azure
Database	MySQL	MySQL Rep	MySQL	SimpleDB	none	DataStore	SQL Azure
Storage / File Sys.	EBS	EC2 & EBS	-	-	S3	GFS	Windows Azure
Consistency	Repeatable Read	Repeatable Read	Repeatable Read	Eventual Consistency	Eventual Consistency	Snapshot Isolation	Snapshot Isolation
App-Language	Java	Java	Java	Java	Java	Java/AppEngine	C#
DB-Language	SQL	SQL	SQL	SimpleDB Queries	low-level API	GQL	SQL
Architecture	**Classic**	**Replication**	**Classic**	**Part.+Repl.**	**Distr. Contol**	**Part.+Repl.(+C)**	**Replication**
HW Config.	manual	manual	manual	manual/automatic	manual	automatic	manual/automatic

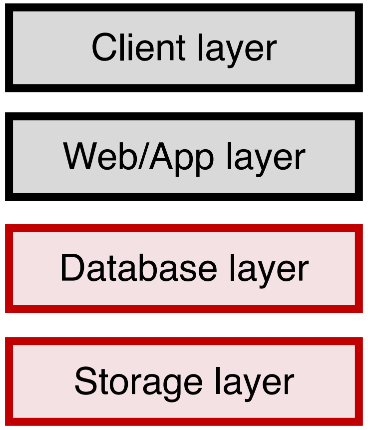
Table 1: Overview of Cloud Services

Single primary bottleneck

Excessive aborts

Architectures Since 2010





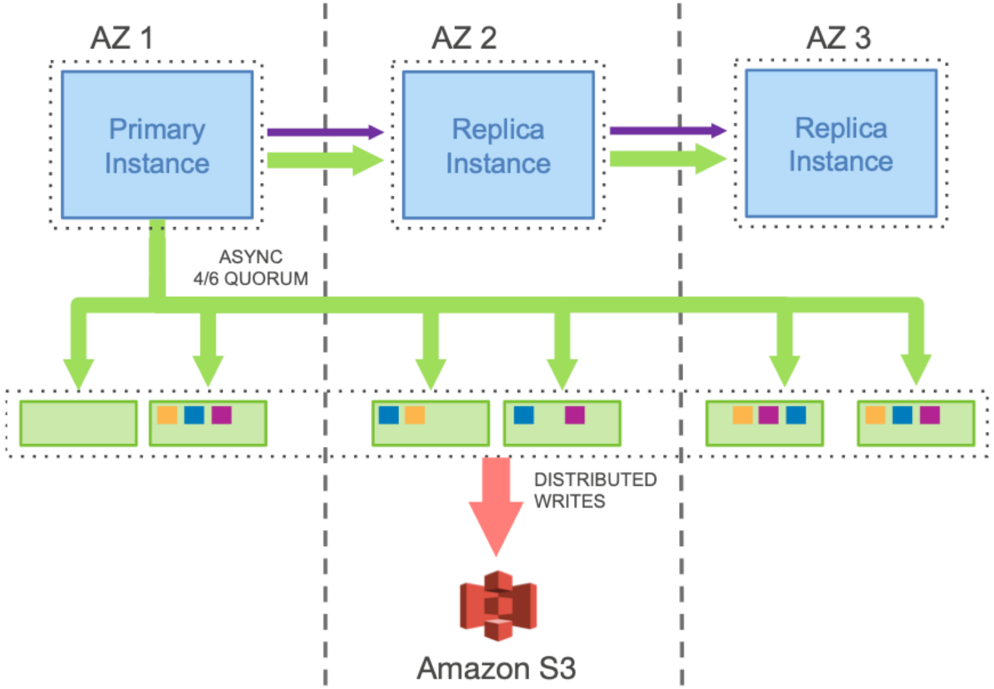
Simple taxonomy

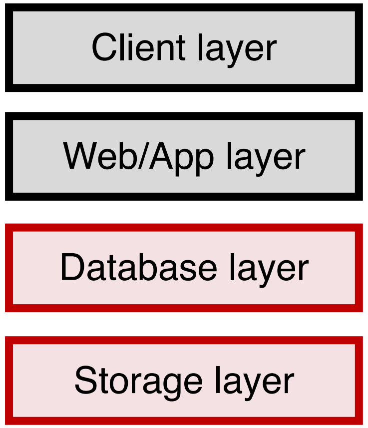
- Integrated or separate
- Single node or multiple nodes
- With or without replication

Google spanner

- Separate DB and storage layer
- Partitioned in the DB layer
- Replication in both layers

Architectures Since 2010





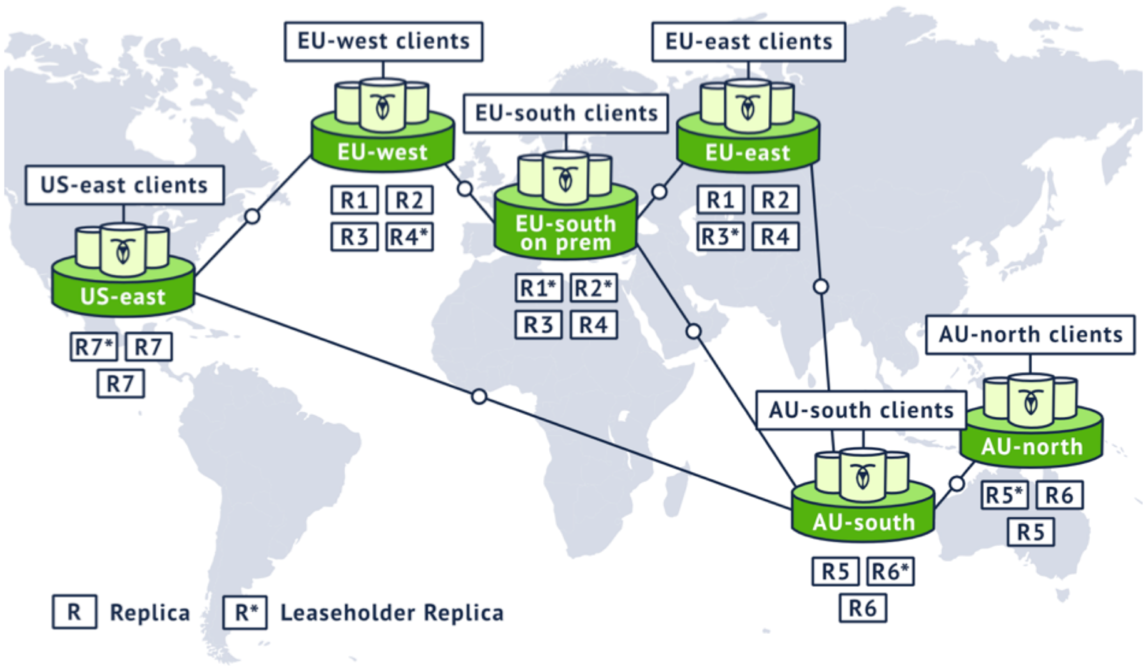
Simple taxonomy

- Integrated or separate
- Single node or multiple nodes
- With or without replication

AWS Aurora

- Separate DB and storage layer
- Single primary in DB layer
- Replication in both layers

Architectures Since 2010



Client layer

Web/App layer

Database layer

Storage layer

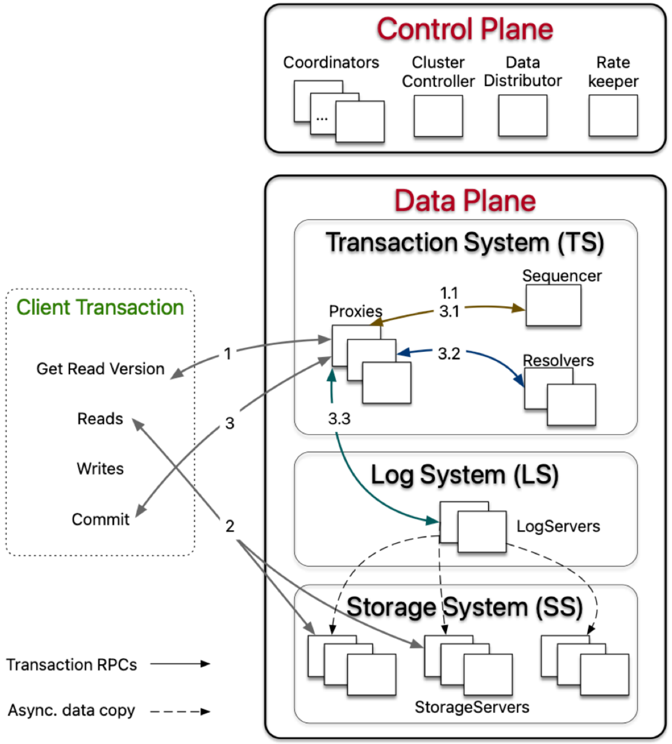
Simple taxonomy

- Integrated or separate
- Single node or multiple nodes
- With or without replication

CockroachDB

- Integrated DB and storage layer
- Partitioned in the DB layer
- With replication

Architectures Since 2010



Client layer

Web/App layer

Database layer

Storage layer

Simple taxonomy

- Integrated or separate
- Single node or multiple nodes
- With or without replication

FoundationDB

- Separate DB and storage layer
- Partitioned in the DB layer
- Replication in log and storage, TS is stateless

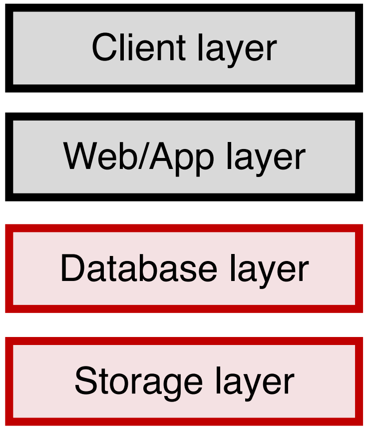
General Trend

Disaggregation of database compute and storage layers

- Aligns with the S3 design in this paper
- More on this topic in the next three lectures

Replication is a must for high availability

Need partitioning in the DB layer for high scalability



Best Cloud-Native DBMS Architecture?

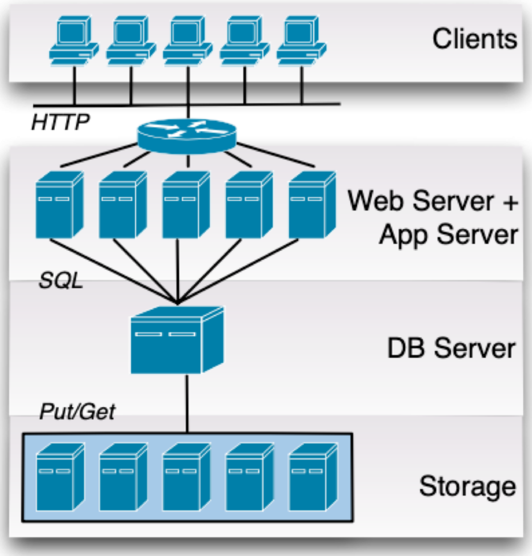


Figure 1: Classic Database Architecture

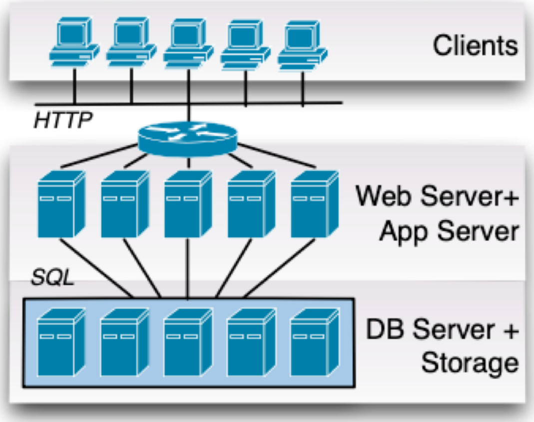


Figure 2: Partitioning

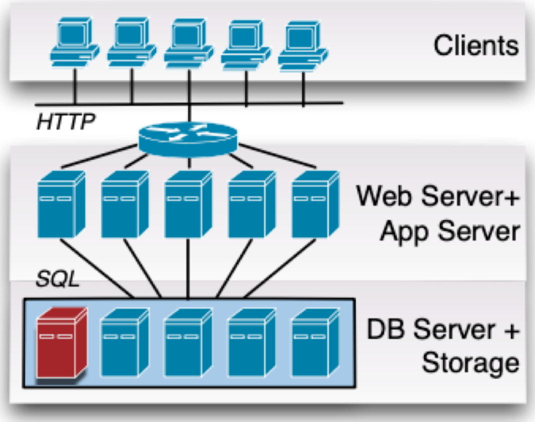


Figure 3: Replication

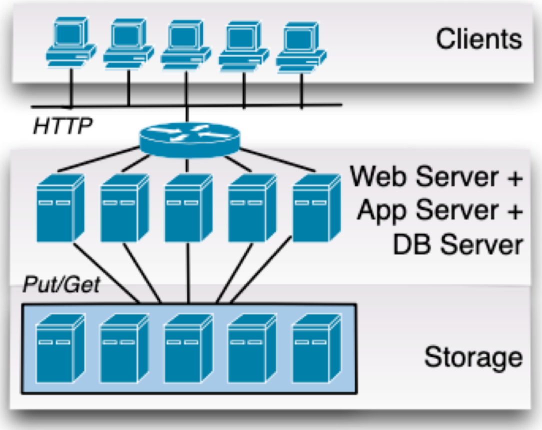


Figure 4: Distributed Control

Best Cloud-Native DBMS Architecture?

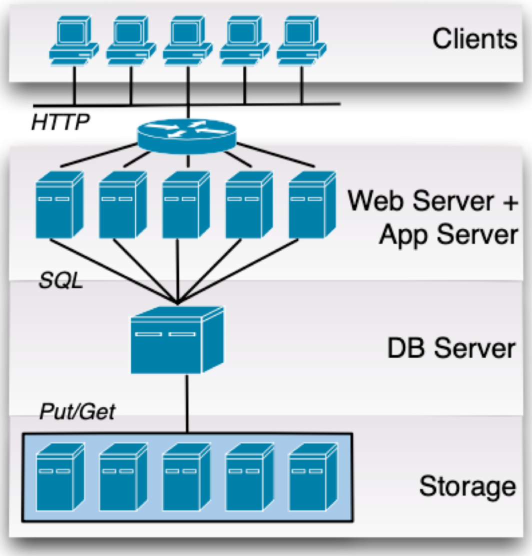


Figure 1: Classic Database Architecture

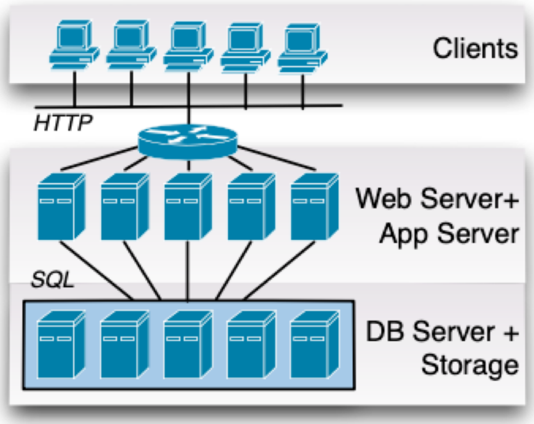


Figure 2: Partitioning

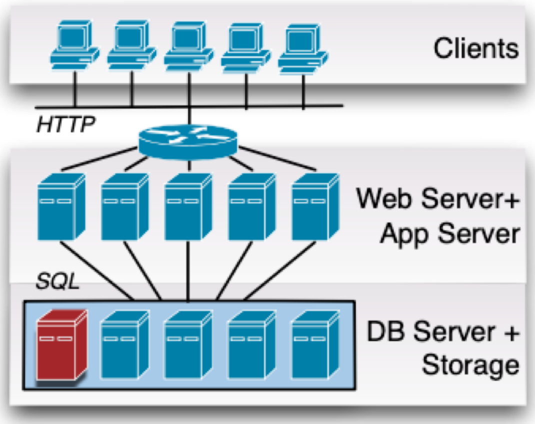


Figure 3: Replication

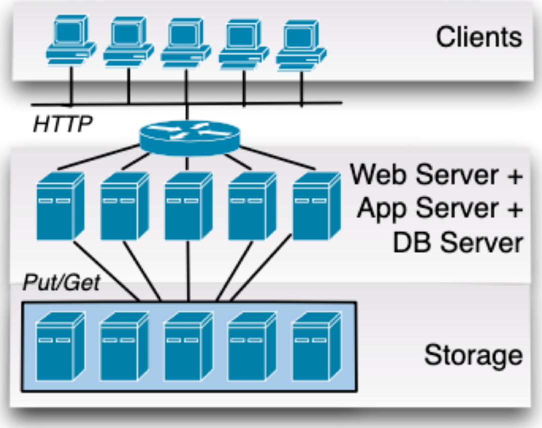
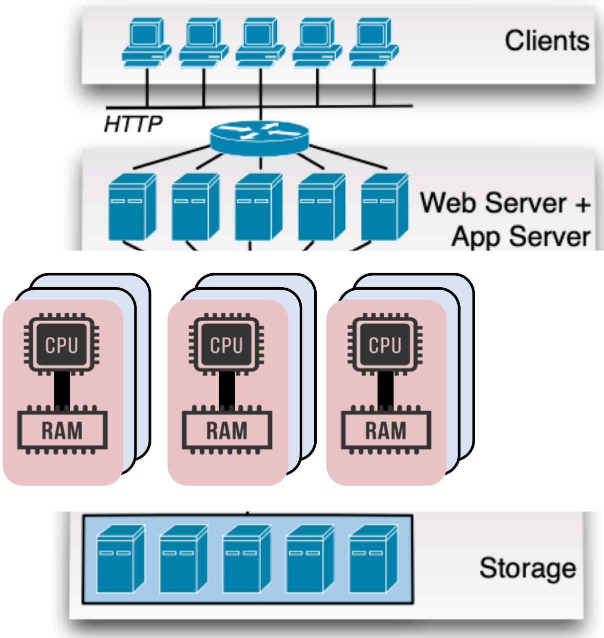


Figure 4: Distributed Control

A prediction:



DB Server
(Partition + replication)

Q/A – Cloud OLTP

How the cache works in these systems? Any consistency issues?

What is a good measure for cost? (bills being paid?)

Implication of storage layer only guaranteeing eventual consistency?

How to create a DB using S3?

What is the best way to solve the scalability issue?

Why not use more than one DB server without partitioning data?

Before Next Lecture

Submit review for

- Alexandre Verbitski, et al., Amazon Aurora: Design Considerations for High Throughput Cloud-Native Relational Databases. SIGMOD, 2017