References

• Amazon SimpleDB Website
  – Part of the Amazon Web services

• Google App Engine Datastore Website
  – Part of the Google App Engine

• Microsoft SQL Azure Website
  – Part of the Azure platform
Motivation

• Running a DBMS is challenging
  – Need to hire a skilled database administrator (DBA)
  – Need to provision machines (hardware, software, configuration)
    • If business picks up, may need to scale quickly
    • In general, workload varies over time

• Solution: Use a DBMS service
  – All machines are hosted in service provider’s data centers
  – Data resides in those data centers
  – Pay-per-use policy
  – Elastic scalability
  – Zero administration

CSE 444 - Fall 2010
Basic Features

- Data storage and query capabilities
- High availability guarantees
- Operations and admin tasks handled by provider

- **Elastic scalability**: Clients pay exactly for the resources they consume; consumption can grow/shrink dynamically
  - No capital expenditures
  - Fast provisioning
Outline

• **Database in the Cloud**
  - DIY Database in the Cloud
  - RDBMS as a Service
  - Database as a Service

• **Discussion**
  - Technical challenges behind databases as a service
  - Broader impacts of databases as a service
Outline: Database in the Cloud

• DIY Database in the Cloud
  – Amazon EC2 + Database software

• RDBMS as a service
• Database as a service
DIY Database in the Cloud

- Setting up a database server
  - Buy land (building, room, server rack)
  - Buy computers and network gears
  - Install the hardware
  - Install OS
  - Install database server

- How can we do this using the cloud?
Amazon Web Services

- Since 2006
- “Infrastructure web services platform in the cloud”

- Amazon Elastic Compute Cloud (Amazon EC2™)
- Amazon Simple Storage Service (Amazon S3™)
- Amazon SimpleDB™
- Amazon CloudFront™
- Amazon Simple Queue Service (Amazon SQS™)
Amazon EC2

- Amazon Elastic Compute Cloud (Amazon EC2™)

- Rent compute power on demand ("server instances")
  - Select required power: small, large, or extra large instance
  - Share resources with other users
  - Variety of operating systems

- Includes: Amazon Elastic Block Store
  - Off-instance storage that persists independent from life of instance
  - Highly available and highly reliable
Amazon S3

• **Amazon Simple Storage Service (Amazon S3™)**
  – “Storage for the Internet”
  – “Web services interface that can be used to store and retrieve any amount of data, at any time, from anywhere on the web.”

• **Some key features**
  – Write, read, and delete uniquely identified objects containing from 1 byte to 5 gigabytes of data each
  – Objects are stored in buckets, located in US or Europe
  – A bucket can be accessed from anywhere
  – Authentication
  – Reliability
Let’s hook up what we’ve got!

• Buy an EC2 instance
• Buy database software
  – Pre-licensed Amazon Machine Image
• Launch your EC2 instance
• Hooray! 😊
  – Took less than 5 minutes to here!
Summary: DIY Database in the Cloud

• Pros
  – Fast provision of resource!
  – Full-control over software
    • Choose your favorite database software!

• Cons
  – Full-control over software
    • Operating system, backup, fault-tolerance, security, performance tuning, scaling, …

• Can cloud provider manage such detail?

CSE 444 - Fall 2010
Outline: Database in the Cloud

• DIY Database in the Cloud

• RDBMS as a service
  – Amazon Relational Database Service
  – Microsoft SQL Azure

• Database as a service
RDBMS as a Service

• Cloud provider manages
  – Operating system, Backup, Fault-tolerance, Scaling

• User manages
  – Schema, Data, Indexes, …

• Use your favorite SQL 😊

• Two services
  – Amazon Relational Database Service
  – Microsoft SQL Azure
Amazon Relational Database Service

• Features
  – Backed by MySQL
  – Easy administration
    • Backup, replication, adding more resources (CPU, disk)
    • Automated or initiated by AWS API or via management console
  – Database size: 5GB ~ 1TB per instance

• How to scale?
  – Adding more resources to master
  – Adding more read-only replicas

CSE 444 - Fall 2010
Microsoft Windows Azure Platform

- "Internet-scale cloud computing and services platform"
- "Provides an operating system and a set of developer services that can be used individually or together"
SQL Azure

- Virtualized SQL Azure Server cluster
  - Azure manages physical cluster

- SQL Azure Server
  - A logical SQL Server
  - Automatically replicated
  - Hosts up to 149 databases
  - 5 GB or 50 GB per database

- How to scale?
  - Partition across multiple databases
    - Automatically distributed across multiple machines
Summary: RDBMS as a Service

• Remove burden of database administration
  – Amazon RDS: automate common administration tasks
  – SQL Azure: separate logical/physical database cluster

• Pros
  – No change to existing applications
  – Support full SQL, transaction, consistencies
  – Highly-available, scalable

• Cons
  – Managing schema, indexes, partitioning, tuning, …
Outline: Database in the Cloud

• DIY Database in the Cloud
• RDBMS as a service

• Database as a service
  – Amazon SimpleDB, Google App Engine
  – Google Fusion Table, Big Query
Database as a Service

• RDBMS as a Service is good
  – But user still has to know a lot about database!

• Can cloud manage following?
  – Schema, Partitioning, Indexing

• Two services
  – Amazon SimpleDB
  – Google App Engine
Amazon SimpleDB

- “Web service providing the core database functions of data indexing and querying”

- **Partitioning**
  - Data partitioned into domains: queries run within domain

- **Schema**
  - No fixed schema
  - Objects are defined with attribute-value pairs
Amazon SimpleDB (2/3)

- **Indexing**
  - Automatically indexes all attributes

- **Support for writing**
  - PUT and DELETE items in a domain

- **Support for querying**
  - GET by key
  - Selection + sort
  - A simple form of aggregation: count
  - Query execution time is limited to 5 second (but can continue)

```
select output_list
from domain_name
[where expression]
[sort_instructions]
[limit limit]
```
Amazon SimpleDB (3/3)

- **Availability and consistency**
  - “Fully indexed data is stored redundantly across multiple servers and data centers”
  - “Takes time for the update to propagate to all storage locations. The data will eventually be consistent, but an immediate read might not show the change”
  - Consistent read, Conditional Put/Delete

- **Integration with other services**
  - “Developers can run their applications in Amazon EC2 and store their data objects in Amazon S3.”
  - “Amazon SimpleDB can then be used to query the object metadata from within the application in Amazon EC2 and return pointers to the objects stored in Amazon S3.”

CSE 444 - Fall 2010
Google App Engine

•  “Run your web applications on Google's infrastructure”

•  Key features
  – Dynamic web serving, with full support for common web technologies: apps serve web requests
  – Persistent storage with queries, sorting and transactions
  – Automatic scaling and load balancing
  – APIs for authenticating users and sending email
  – A fully featured local development environment that simulates Google App Engine on your computer

•  Limitation: apps must be written in Python or Java
Google App Engine Datastore (1/3)

• “Distributed data storage service that features a query engine and transactions”

• **Partitioning**
  – Data partitioned into “entity groups”
  – Entities of the same group are stored together for efficient execution of transactions

• **Schema**
  – Each entity has a key and properties that can be either
    • Named values of one of several supported data types (includes list)
    • References to other entities
  – Flexible schema: different entities can have different properties
Google App Engine Datastore (2/3)

• **Indexing**
  – Applications define indexes: must have one index per query type

• **Support for writing**
  – PUT and DELETE entities

• **Support for querying**
  – Fetch an entity using its key
  – Execute a query: selection + sort
  – Language bindings: either invoke methods or write GQL
  – Lazy query evaluation: query executes when user accesses results
Google App Engine Datastore (3/3)

- **Availability and consistency**
  - Every datastore write operation (put/delete) is atomic
  - Support transactions
    - All operations must operate on entities in the same entity group
    - Cannot perform queries; can only get entities by their keys
  - Optimistic concurrency control
Summary: Database as a Service

- **Partitioning**: in all systems data is partitioned
- **Schema**: flexible schema
  - Different entities can have different attributes
- **Indexing**: all systems answer queries using indexes
- **Write** operations: put and delete
  - Some systems support transactions on objects within a group
- **Query** interface: primarily selection + sort
- **Availability and consistency**
  - All systems strive to achieve high availability
  - Some systems have strong consistency others weak
## Comparison

<table>
<thead>
<tr>
<th></th>
<th>Hardware Admin.</th>
<th>System Admin.</th>
<th>Database Admin.</th>
<th>Scale</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traditional</td>
<td>High</td>
<td>High</td>
<td>High</td>
<td>Hard</td>
<td>High</td>
</tr>
<tr>
<td>DIY in the Cloud</td>
<td>None</td>
<td>High</td>
<td>High</td>
<td>Moderate</td>
<td>High</td>
</tr>
<tr>
<td>RDBMS as a Service</td>
<td>None</td>
<td>~ Low</td>
<td>Medium</td>
<td>~ Easy</td>
<td>High</td>
</tr>
<tr>
<td>Database as a Service</td>
<td>None</td>
<td>None</td>
<td>~ Low</td>
<td>Easy</td>
<td>~ High?</td>
</tr>
</tbody>
</table>
Related Database as a Service

• Google Fusion Table
  – Share your data with rest of the world!
  – Easy visualization, SQL-like query language

• Google Big Query
  – Run SQL-like query over TB or PB of data in seconds!
  – Currently in preview
Outline

• **Database in the Cloud**
  – DIY Database in the Cloud
  – RDBMS as a Service
  – Database as a Service

• **Discussion**
  – Technical challenges behind databases as a service
  – Broader impacts of databases as a service
Challenges of DBMS as a Service

- Scalability requirements
  - Large data volumes and large numbers of clients
  - Variable and heavy workloads
- High performance requirements: interactive web services
- Consistency and high availability guarantees
- Service Level Agreements
- Security
- Billing
- Loading/Exporting Data
Broader Impacts

• Cost-effective solution for building web services

• Content providers focus only on their application logic
  – Service providers take care of administration
  – Service providers take care of operations

• Security/privacy concerns: all data stored in data centers