Introducing Datastore

Google App Engine

Google Cloud Platform

Agenda

1. What Is Datastore ?
2. Kinds, Entity, and Keys
3. Saving and Getting Entities
4. Exercise

Cloud Datastore

NoSQL Store

Designed to store **billions of rows**, across **thousands of machines**, replicated across **multiple datacenters**

Schemaless access

Schemaless access, no need to think about underlying data structure. Idiomatic client libraries for simple development

Autoscale and management

Automatically scales as your needs do and is fully managed. **Sharding** and **replication** is taken care of for you

Cloud Datastore

- Datastore is a database (persistent storage) for App Engine

	App Engine	Traditional Web Applications
Web application framework	App Engine (Java, Python, Go)	Perl/CGI PHP Ruby on Rails, etc
Persistent storage	Datastore	RDBMS • MySQL • PostgreSQL • SQL Server • Oracle

Cloud Datastore

- Datastore vs. RDBMS

	Datastore	RDBMS
Query language flexibility	SQL-like query language • Limited to simple filter and sort	Full support of SQL • Table JOIN • Flexible filtering • Subquery
Reliability and Scalability	Highly scalable and reliable	Harder to scale

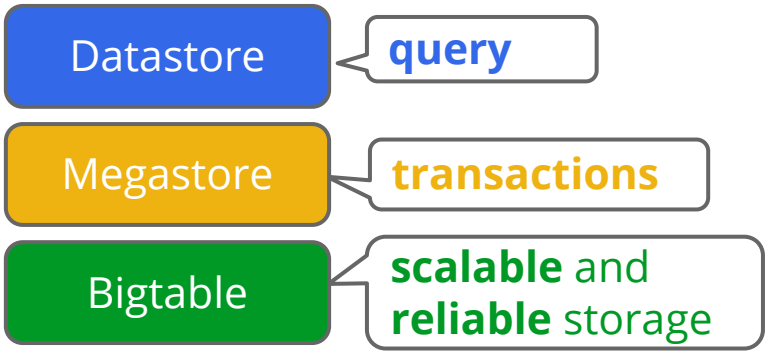
- Datastore offers ‘Google-level’ reliability and scalability

Strong Consistency vs. Eventual Consistency

Strong Consistency	Data is always consistent among all database instances • Read after write operation • Even if crash in the middle of write operation
Eventual Consistency	Takes time until all data becomes consistent after write (Think of DNS as an example)

Datastore Internals

- Based on Bigtable, which offers **super high scalability**
- High **availability** by High Replication Datastore (**HRD**)
  - Synchronous write on multiple datacenters
- Supports **strong consistency**



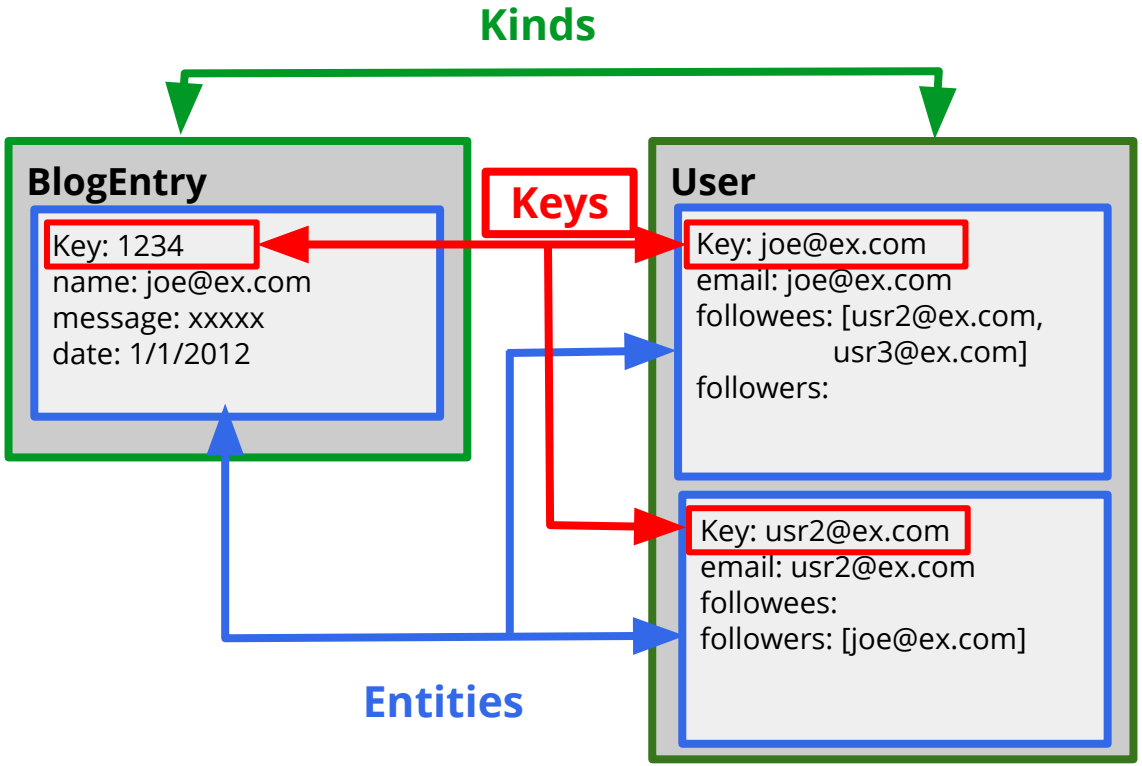
Kinds, Entities, and Keys

Basic Terminology

- Different terms for corresponding concepts

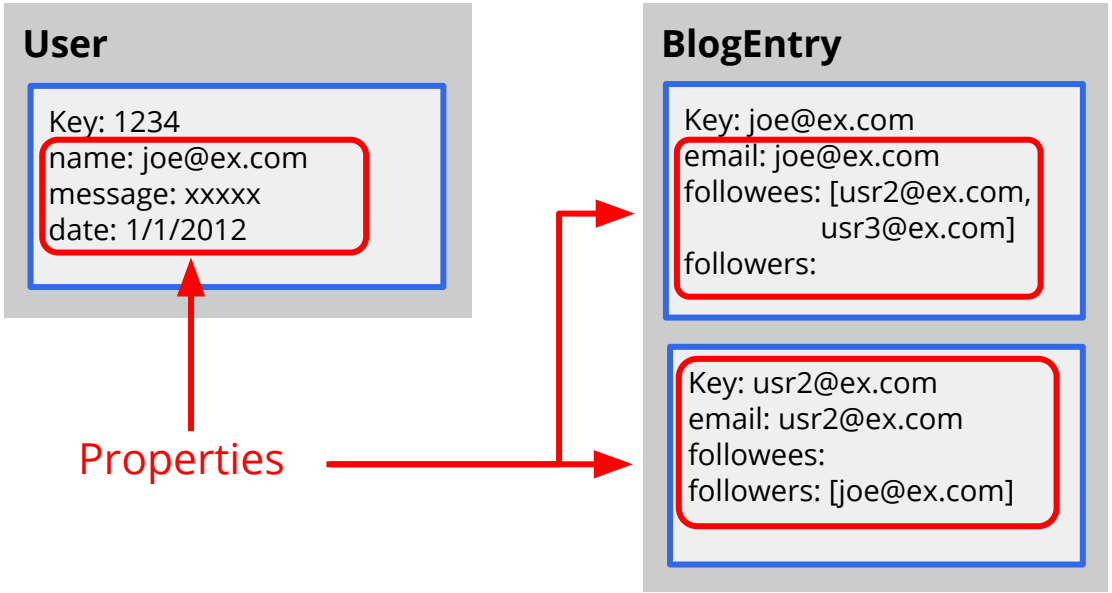
	Datastore	RDBMS
Category of object	Kind	Table
One entry/object	Entity	Row
Unique identifier of data entry	Key	Primary Key
Individual data	Property	Field

Kind, Entity, and Key



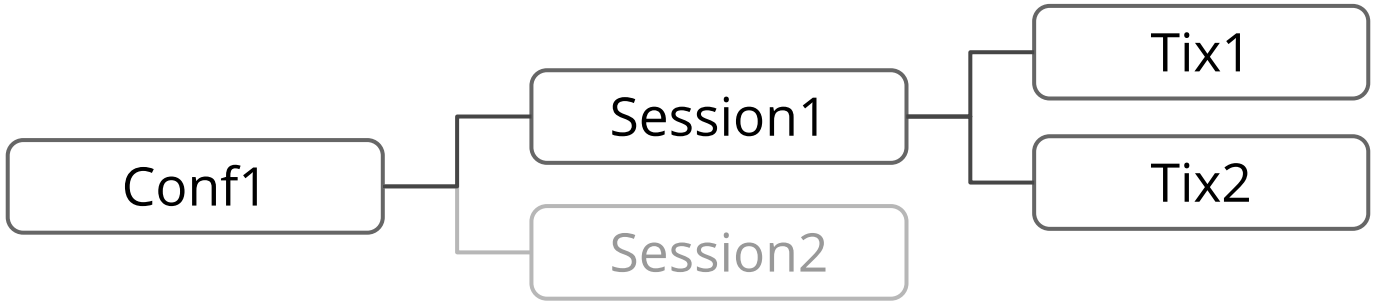
Properties and Data Types

- Each entity has one or more named properties
  - Variety of data types (int, float, boolean, String, Date etc)
  - Can be multi-valued



Entity Groups

An entity can have a parent

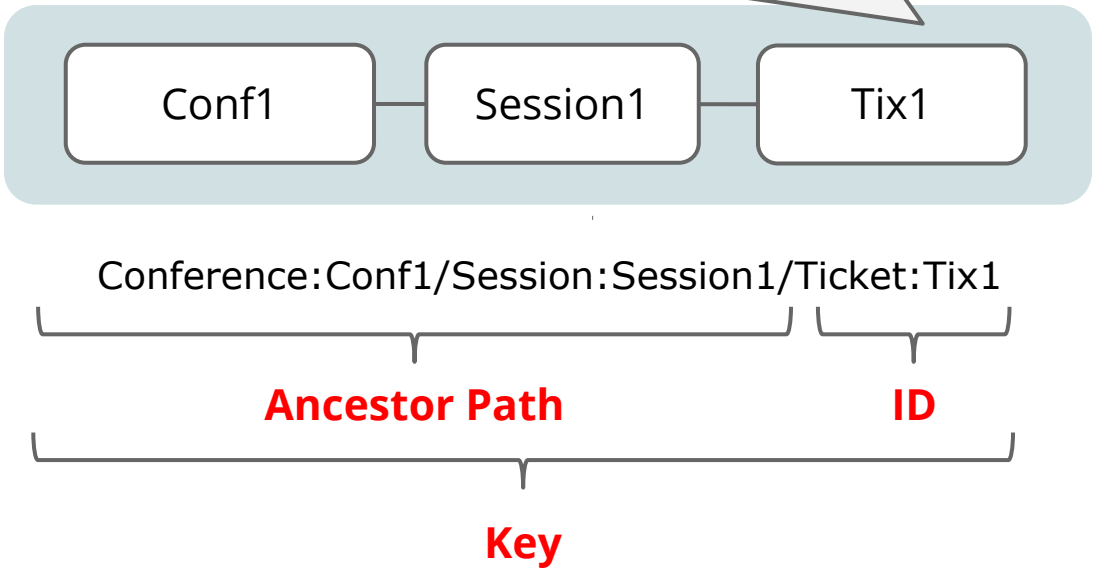


- ➔ Hierarchy of entities stemming from a root entity is an ***Entity Group***
- ➔ Each entity is its own entity group by default
- ➔ Parent child relationships are forever!

Entity Key Includes Ancestor Path

- Specify parent's Key when creating a new entity

```
Entity Tix1 = new Entity("Ticket", "Tix1", Session1.getKey());
```



Saving and Getting Entities

Datastore API

Java

- Low-level API
  - The best performance, but more coding
  - JDO/JPA
    - More portability by Java standard API
  - Third party frameworks
    - Objectify, Twig, Slim3 ...
    - Sophisticated features with better performance

Python

- DB API
  - Traditional Datastore API for Python
- NDB API (New DB)
  - Automatic entity caching (memcache), sophisticated queries, atomic transactions

Creating an Entity

Java

```
DatastoreService datastore =
    DatastoreServiceFactory.getDatastoreService();

Entity employee = new Entity("Employee");
employee.setProperty("name", "Antonio Salieri");
employee.setProperty("hireDate", new Date());
Key empKey = datastore.put(employee);
```

Creating an Entity

Python

```
class Employee(db.Model):
    name = db.StringProperty(required=True)
    hire_date = db.DateProperty()

e = Employee(name="Antonio Salieri")
e.hire_date = datetime.datetime.now().date()
e_key = e.put()
```

Getting an Entity

Java

```
// Use email as key when creating entity
Entity employee = new Entity("Employee", "joe@");
datastore.put(employee);

// Later, use the key to retrieve the entity
Key userKey = KeyFactory.createKey( "Employee", "joe@" );
Entity user = datastore.get(userKey);
```

Getting an Entity

Python

```
// Use email as key when creating entity
employee = Entity(key_name="joe@");
employee.put();

user_key = db.Key.from_path('Employee', 'joe@)
user = db.get(user_key)
```

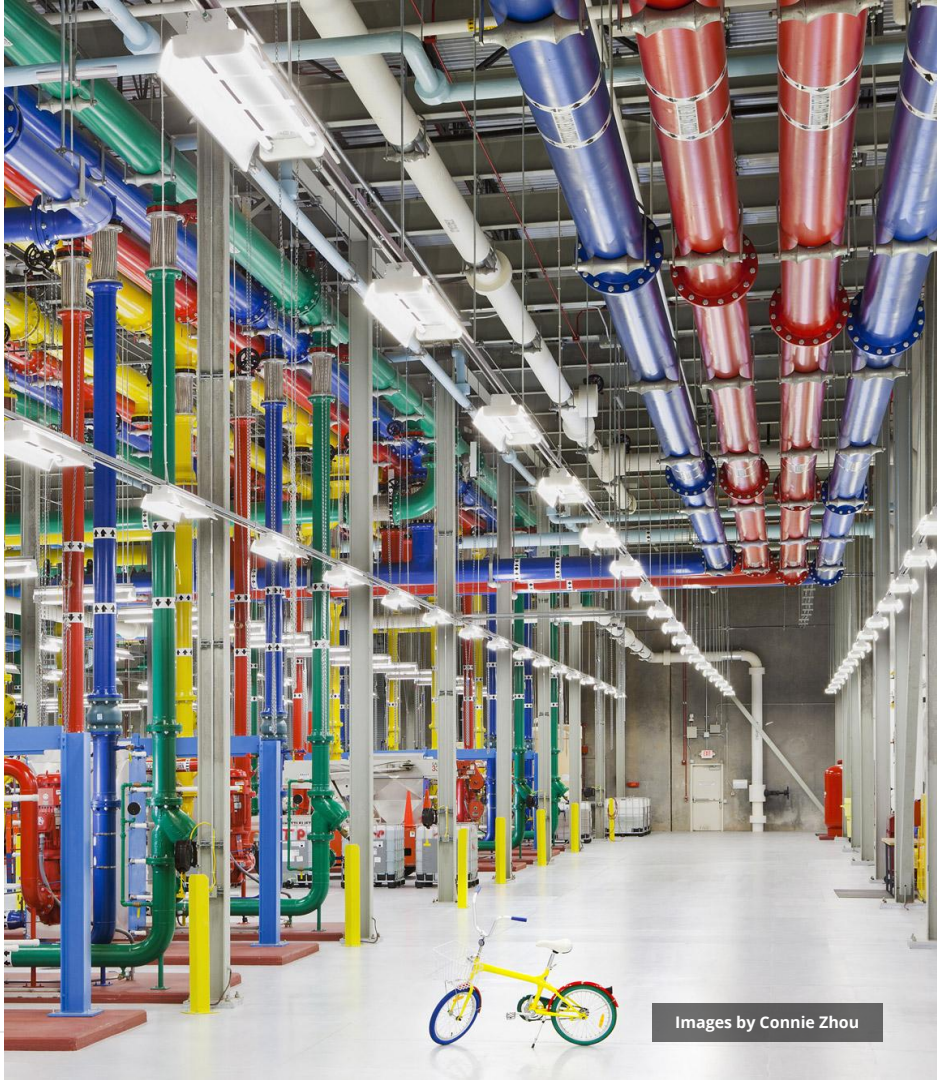
Quiz

A datastore **‘Entity’** corresponds to which of the following RDBMS concepts:
*(pick **one** answer)*

- ❑ Field
- ❑ Kind
- ❑ Table
- ❑ Primary Key
- ❑ Row
- ❑ None of the above

Hands-on

- You will create an App Engine application in either Java or Python
- You will then integrate the Datastore API to understand how it works



Resources

- Developer Documentation:
  - Storing Data (Java) (Python)
  - Datastore Overview (Java) (Python)
  - Entities, Properties and Keys (Java) (Python)

Articles:

- Life of a Datastore write


cloud.google.com
Images by Connie Zhou