
Google
Developers

No-SQL vs. SQL

Battle of the Backends

Alfred Fuller
Ken Ashcraft

Data in the Cloud

Why Cloud?

- Fault Tolerance
  - We man the pagers for you
  - Automated failure recovery
- Low maintenance
  - We manage updates on every level for you (bare metal -> software patches)
  - Focus on what you do best
- Durability
  - Built-in replication
  - Distributed geographically
- Accessibility
  - Always on, always available (as long as you have an internet connection)
  - Local development environments

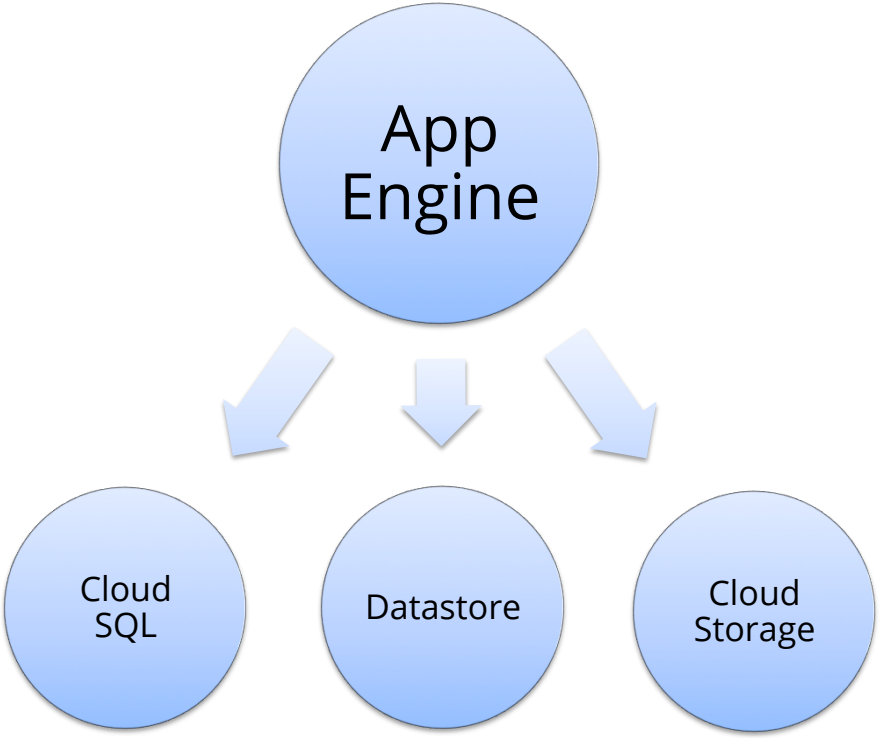
Google App Engine (GAE)

- Build apps on Google's infrastructure

- Platform as a Service (PaaS)
  - Easy to build
  - Easy to scale
  - Easy to maintain

- Focus on what makes your app great!

App Engine + Storage



App Engine Datastore

- Google storage infrastructure
- Same technology we use for our own applications
- Distilled into well documented APIs
- Built for scale (size and traffic)
  - 2 Trillion operations per month
- Fully managed ‘NoSQL’ solution

Cloud SQL

- Fully managed
- Pure MySQL

No-SQL vs. SQL

Battle of the Backends

Alfred Fuller
Ken Ashcraft

Queries

No-SQL?

- We support an ever growing subset of SQL
  - Filters
    - SELECT * FROM Table WHERE A=1 AND (B=2 **OR** C=3)
  - Sorting
    - SELECT * FROM Table ORDER BY A, B DESC
  - Projections / Index-Only Queries
    - SELECT **A, B** FROM Table
- Beyond SQL
  - Repeated properties
    - Contains all(==) / any(IN)
- Scales in the size of the result set!

Aggregation

“Compute the average age of people in each city.”

Aggregations

"Compute the average age of people in each city."

SQL

```
SELECT people.city_id, AVG(people.age)
  FROM people
  GROUP BY people.city_id;
```

MapReduce

Map			Shuffle				Reduce
city=3, age=5	3	5					
city=1, age=2	1	2					
city=3, age=7	3	7	1	2	3		5 / 2 = 2.5
city=4, age=9	4	9	2	3			3 / 1 = 3
city=4, age=9	4	9	3	5	7		12 / 2 = 6
city=1, age=3	1	3	4	9	9	8	26 / 3 = 8.66
city=4, age=8	4	8					
city=2, age=3	2	3					

Person

"Compute the average age of people in each city."

Materialized View

Track Changes

city=**4**, age=5

city=**4**, age=2

city=**1**, age=7

city=4, age=9

city=4, age=9

city=1, age=3

city=4, age=8

city=**4**, age=3

Fan-in and Apply

1	-2,-1	+7,+1	
2	-3,-1		
3	-7,-1	-5,-1	
4	+2,+1	+3,+1	+5,+1

1	10 / 2 = 5
2	3 / 1 = 3
3	12 / 2 = 6
4	36 / 6 = 6

“Compute the average age of people in each city.”

Joins

"Compute the average age of people in each city and look up the location for that city."

SQL

```
SELECT AVG(people.age), cities.name, cities.latitude, cities.longitude
  FROM people, cities
  WHERE people.city_id = cities.city_id
  GROUP BY people.city_id;
```

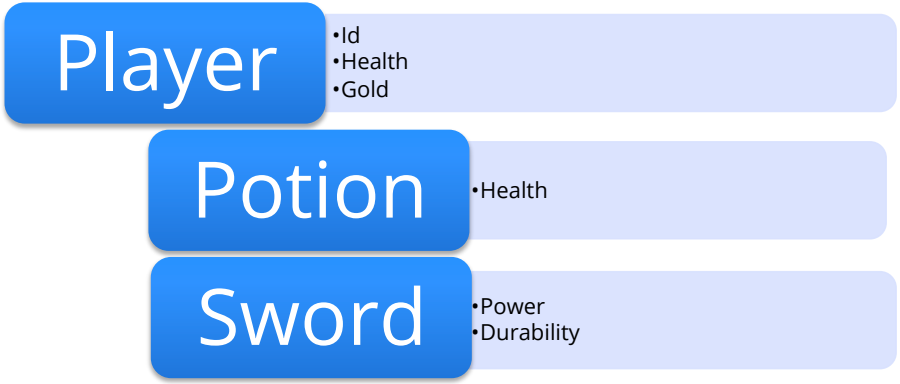
Scoreboard

	Datastore	Cloud SQL
Queries	✔	✔+

Transactions

Entity Groups

- Grouping of entities under a single transaction log
- Many entity groups = scalable ACID semantics



Multi-row transactions

Python

```python
@db.transactional
def use_potion():
  player1 = get_player(1)
  potion = player1.get_item("potion")
  player1.health += potion.health
  db.delete(potion)
  db.put(player1)
```

Player
- Id
- Health
- Gold

Potion
- Health

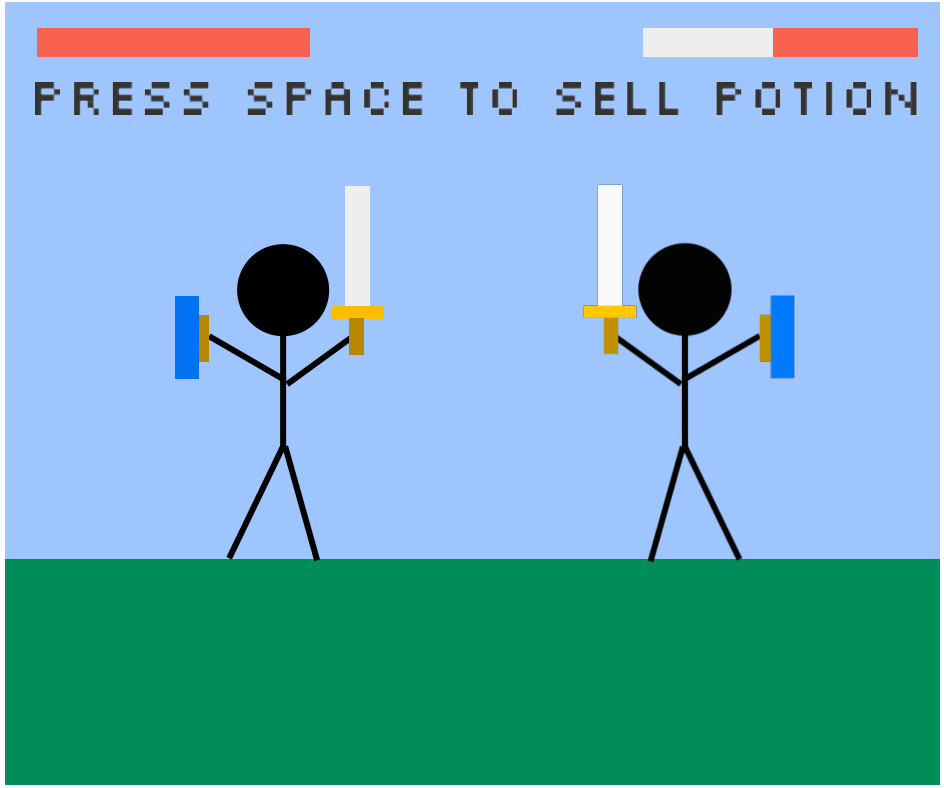
XG transactions

Python

```python
db.transactional(xg=true)
def sell_potion(id1, id2):
  buyer = get_player(id1)
  seller = get_player(id2)
  potion = seller.get_item("potion")
  seller.gold += 25
  buyer.gold -= 25
  buyer.store_item(potion)
  db.delete(potion)
  db.put(buyer, seller)
```

Player
- Id
- Health
- Gold

Potion
- Health



Transactions in SQL

- “Sell a potion to another player”

SQL

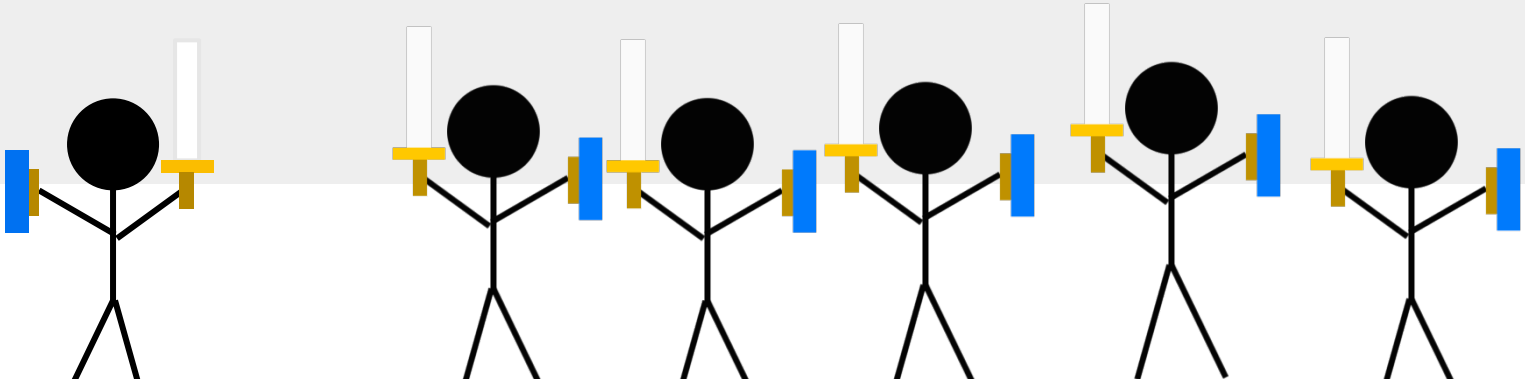
```
START TRANSACTION;
SELECT gold FROM players WHERE id IN (1, 2);
SELECT COUNT(*) FROM inventory WHERE player_id = 1 AND type = ‘potion’;
UPDATE players SET gold = gold + 25 WHERE id = 1;
UPDATE players SET gold = gold – 25 WHERE id = 2;
UPDATE inventory SET player_id = 2 WHERE player_id = 1 AND type = ‘potion’ LIMIT 1;
COMMIT;
```

Transactions in SQL

- "Give gold to all of your friends"

SQL

```
START TRANSACTION;
SELECT gold FROM players WHERE id = 1;
SELECT COUNT(*) FROM friends WHERE player_id = 1;
UPDATE players SET gold = <amount to give away>  WHERE id = 1;
UPDATE players, friends SET players.gold = players.gold + 25
  WHERE friends.player_id = 1 AND players.id = friends.friend_id;
COMMIT;
```



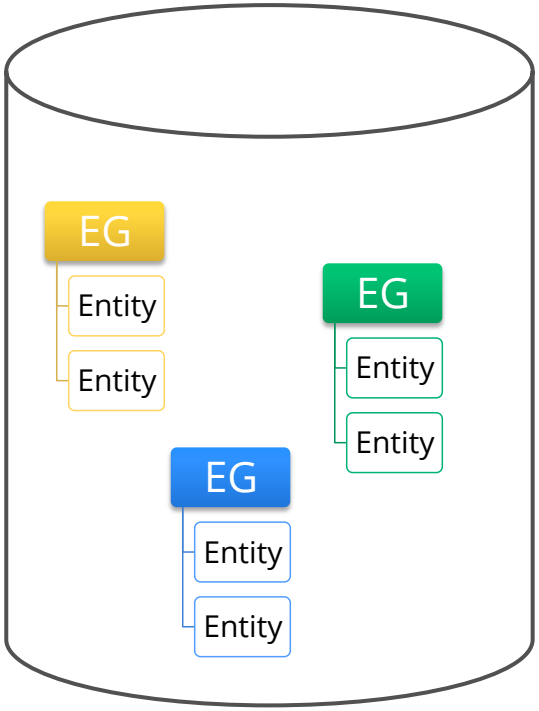
Scoreboard

	Datastore	Cloud SQL
Queries	✔	✔+
Transactions	✔	✔+

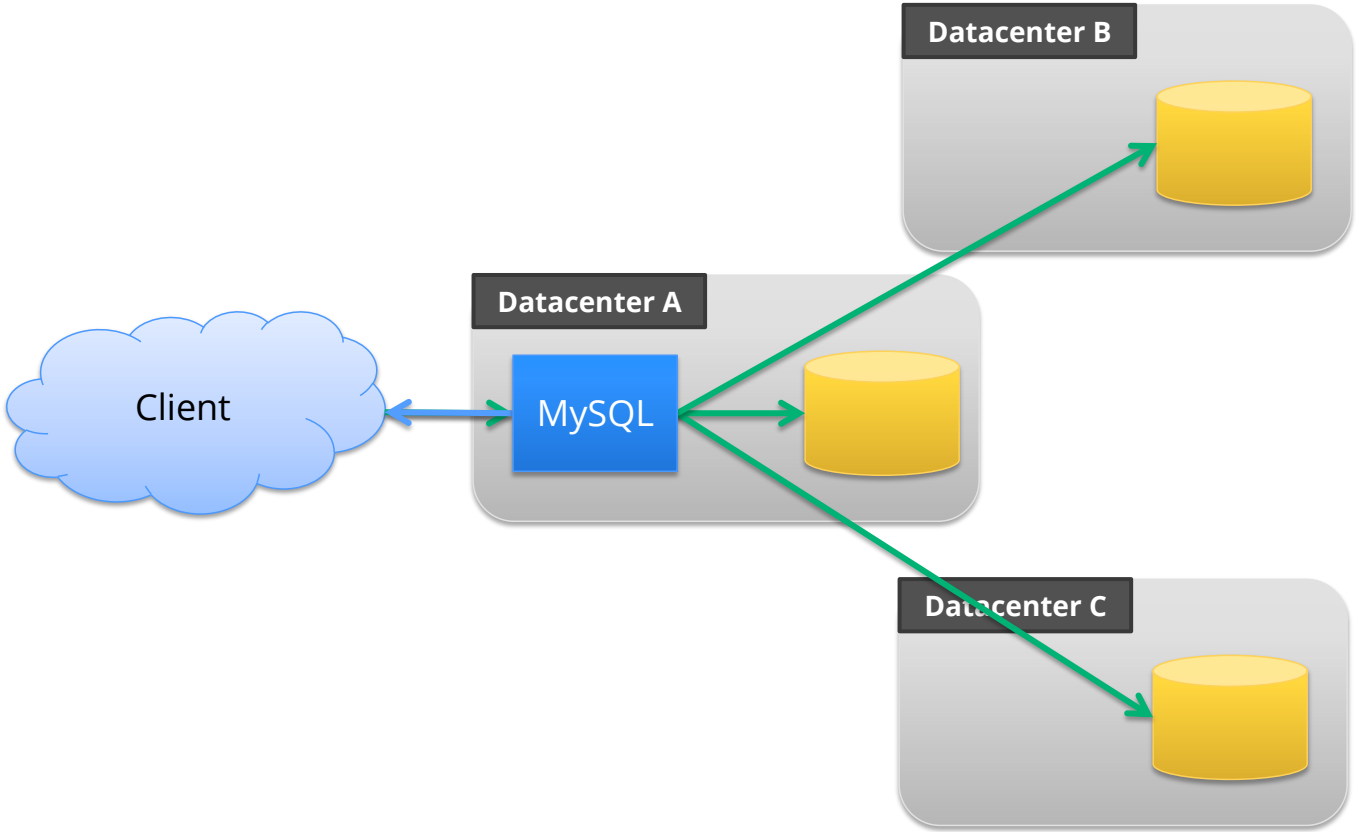
Consistency

Consistency

- Megastore Replication!
- Entity groups
  - Parallel transaction logs
  - Parallel replication
- No Master
- Strong within an entity group
  - Get
  - Ancestor Query
- Eventual across entity groups
  - Global Queries



Master + Synchronous Replication



Scoreboard

	Datastore	Cloud SQL
Queries	✔	✔+
Transactions	✔	✔+
Consistency	✔	✔+

Scalability

Cloud SQL

Google Time Keeper

- Used by Google AdWord's sales and support team
- Tracks time spent on
  - Chat support
  - Email support
  - Campaign optimization

Google Org Chart

- Tracks 30k+ employees
- 10-100 QPS

Disgruntled Pigeons

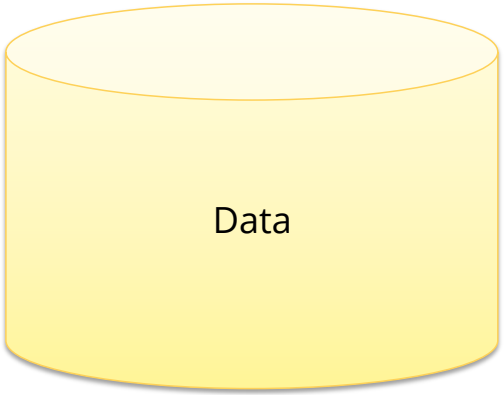
- Thousands of QPS
- Millions of users
- Billions of ruffled feathers

Datastore on Megastore on Bigtable on ...

- All the best features of each layer

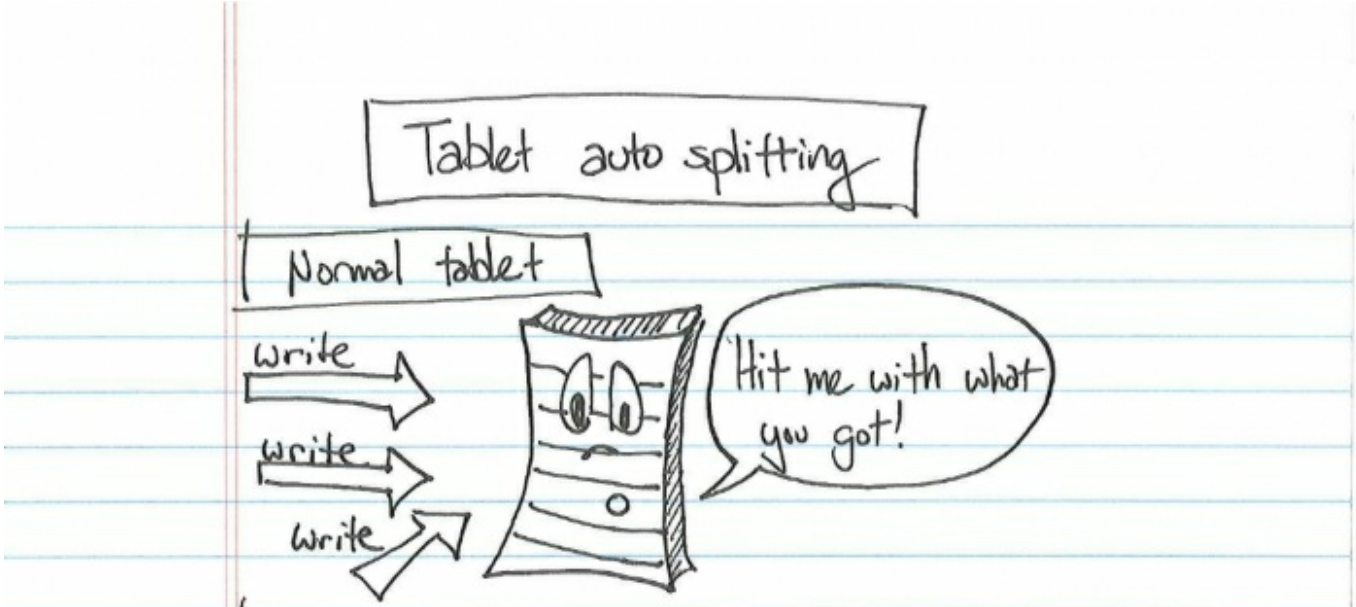
GFS v2

- Huge Capacity
- Durable



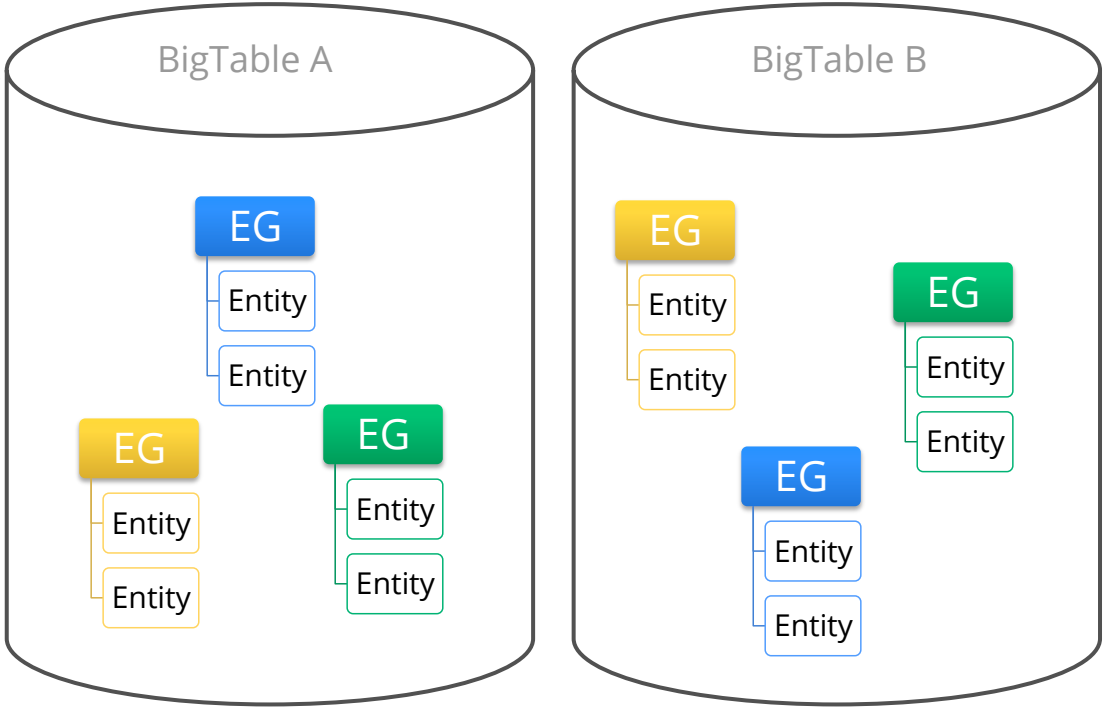
BigTable Load Balancing

- Automatically splits and balances data based on load
- Scales linearly with available resources



Megastore

- Works at scale
  - See 2011 talk “More 9s Please: Under The Covers of the High Replication Datastore”
  - 9’s are important at scale
- Not reliant on a single datacenter
- Handles local issues
- Handles catastrophic failures



Scoreboard

	Datastore	Cloud SQL
Queries	✔	✔+
Transactions	✔	✔+
Consistency	✔	✔+
Scalability	✔+	✔

Management

Create an Application

You have 8 applications remaining.

Application Identifier:

.appspot.com Check Availability

All Google account names and certain offensive or trademarked names may not be used as Application Identifiers.
You can map this application to your own domain later. Learn more

Application Title:

Displayed when users access your application.

Authentication Options (Advanced): Learn more

Google App Engine provides an API for authenticating your users, including **Google Accounts**, **Google Apps** , and **OpenID**. If you choose to use this feature for some parts of your site, you'll need to specify now what type of users can sign in to your application:

Open to all Google Accounts users (default)

If your application uses authentication, anyone with a valid Google Account may sign in.

Edit

Storage Options (Advanced):

Google App Engine datastore options.

High Replication (default)

Uses a more highly replicated Datastore that makes use of a system based on the Paxos algorithm to synchronously replicate data across multiple locations simultaneously. Offers the highest level of availability for reads and writes at the cost of eventual consistency for some queries. Note: High Replication Datastore is required in order to use the Python 2.7 and Go runtimes.

Edit

Create Application Cancel

Create an Application

You have 8 applications remaining.

Application Identifier:

sql-vs-nosql .appspot.com Check Availability

All Google account names and certain offensive or trademarked names may not be used as Application Identifiers.
You can map this application to your own domain later. Learn more

Application Title:

SQL vs NoSQL

Displayed when users access your application.

Authentication Options (Advanced): Learn more

Google App Engine provides an API for authenticating your users, including **Google Accounts**, **Google Apps** , and **OpenID**. If you choose to use this feature for some parts of your site, you'll need to specify now what type of users can sign in to your application:

Open to all Google Accounts users (default)

If your application uses authentication, anyone with a valid Google Account may sign in.

Edit

Storage Options (Advanced):

Google App Engine datastore options.

High Replication (default)

Uses a more highly replicated Datastore that makes use of a system based on the Paxos algorithm to synchronously replicate data across multiple locations simultaneously. Offers the highest level of availability for reads and writes at the cost of eventual consistency for some queries. Note: High Replication Datastore is required in order to use the Python 2.7 and Go runtimes.

Edit

Create Application Cancel

Application Registered Successfully

The application will use **sql-vs-nosql** as an identifier. This identifier belongs in your application's configuration as well. Note that this identifier cannot be changed. Learn more

The application uses the **High Replication** storage scheme. Learn more

If you use Google authentication for your application, **SQL vs NoSQL** will be displayed on Sign In pages when users access your application.

Choose an option below:

- View the dashboard for SQL vs NoSQL.
- Use appcfg to upload and deploy your application code.
- Add administrators to collaborate on this application.

Google app engine

[redacted]@gmail.com | My Account | Help | Sign out

sql-vs-nosql [High Replication] | No version deployed!

Report Production Issue « My Applications

Main
- Dashboard
- Instances
- Logs
- Versions
- Backends
- Cron Jobs
- Task Queues
- Quota Details

Data
- Datastore Indexes
- Datastore Viewer
- Datastore Statistics
- Blob Viewer
- Prospective Search
- Text Search
- Datastore Admin
- Memcache Viewer

Administration
- Application Settings
- Permissions
- Blacklist
- Admin Logs

You need to upload and deploy an application before you can make Web history.
Read about using appcfg to upload and deploy one.

Charts

No Data Available

Instances

Number of Instances - Details	Average QPS	Average Latency	Average Memory

Billing Status: Free - Settings

Quotas reset every 24 hours. Next reset: 14 hrs

Resource	Usage

Current Load

URI	Requests last 10 hrs	Runtime MCycles last hr	Avg Latency last hr

Errors

URI	Count	% Errors last 10 hrs

Google apis

SQL vs NoSQL

- Overview
- Services
- Team
- API Access
- Billing
- Google Cloud SQL

Dashboard

Project Summary	
Name	SQL vs NoSQL
Project ID	Register...
Owners	[redacted]@gmail.com - you
Current charges	Click here to administer your billing settings...

Google apis

SQL vs NoSQL

- Overview
- Services
- Team
- API Access
- Billing
- Google Cloud SQL

Billing

Billing is not enabled Learn more

Enable billing

Google Checkout

Unbilled usage (estimate, updated daily)

Start date

Total (before taxes)

Statements

None

Google apis

SQL vs NoSQL

- Overview
- Services
- Team
- API Access
- Billing
- Google Cloud SQL

Dashboard

Project Summary	
Name	SQL vs NoSQL
Project ID	Register...
Owners	[redacted]@gmail.com - you
Current charges	None

Service	Status
Google Cloud SQL	No known issues

Google apis

SQL vs NoSQL

- Overview
- Services
- Team
- API Access
- Billing
- Google Cloud SQL

Overview

New instance...

Overview

Create a new instance to get started.

Learn more about Google Cloud SQL

Google apis

SQL vs NoSQL

- Overview
- Services
- Team
- API Access
- Billing
- Google Cloud SQL

Overview

New instance...

Overview

New Instance

Name: Required

Size: D1

Pricing plan: Per use

Learn more about pricing and instance sizes

Authorized applications

1. App Engine application ID

Create instance Cancel

Code Home - Privacy Policy

Google apis

SQL vs NoSQL

- Overview
- Services
- Team
- API Access
- Billing
- Google Cloud SQL

Overview

New instance...

Overview

started.

d SQL

New Instance

Name: sql-is-better

Size: D1

Pricing plan: Per use

Learn more about pricing and instance sizes

Authorized applications

1. App Engine application ID

Create instance | Cancel

Google apis

SQL vs NoSQL

- Overview
- Services
- Team
- API Access
- Billing
- Google Cloud SQL

Overview

New instance...

Overview

started.

d SQL

New Instance

Name: sql-is-better

Size: D1

Pricing plan: Per use

Learn more about pricing and instance sizes

Authorized applications

1. sql-vs-nosql
2. App Engine application ID

Create instance | Cancel

Google apis

SQL vs NoSQL

- Overview
- Services
- Team
- API Access
- Billing
- Google Cloud SQL

Overview

New instance...

Overview

Register Project ID

A project ID is a unique, DNS-compatible label similar to a hostname that is used by certain services to locate your project and access its resources. A project ID is only required when a service you use depends on it.

Your ID will be globally unique.

Once a project ID has been registered, it cannot be changed.

Project ID: Check availability

6–63 lowercase letters, digits, or hyphens. Must start with a letter. Trailing hyphens are prohibited.

Choose this ID | Cancel

Create instance | Cancel

Code Home - Privacy Policy

Google apis

SQL vs NoSQL

- Overview
- Services
- Team
- API Access
- Billing
- Google Cloud SQL

Overview

New instance...

Overview

Register Project ID

A project ID is a unique, DNS-compatible label similar to a hostname that is used by certain services to locate your project and access its resources. A project ID is only required when a service you use depends on it.

Your ID will be globally unique.

Once a project ID has been registered, it cannot be changed.

Project ID: sql-is-better [Check availability] ID sql-is-better is available!

6–63 lowercase letters, digits, or hyphens. Must start with a letter. Trailing hyphens are prohibited.

[Choose this ID] [Cancel]

[Create instance] [Cancel]

Google apis

SQL vs NoSQL

- Overview
- Services
- Team
- API Access
- Billing
- Google Cloud SQL

Creating instance sql-is-better:sql-is-better.

Dashboard | Logs | SQL Prompt | Backups

Instance settings | Actions | New instance...

Dashboard for Instance sql-is-better:sql-is-better

Properties

Status:	Being created
Version:	MySQL 5.5
Size:	D1
Pricing Plan:	Per use
Replication Type:	Synchronous
Disk Usage:	unknown / 10 GB

Authorized Applications

sql-vs-nosql

Storage Usage (GB) | Month | Day | Hour | Minute

All times are UTC.

Zoom: 1d 5d 1m 3m 6m 1y Max

Count 0 | June 01, 2012

Google apis

SQL vs NoSQL

- Overview
- Services
- Team
- API Access
- Billing
- Google Cloud SQL

Dashboard | Logs | SQL Prompt | Backups | Instance settings | Actions | New instance...

Dashboard for Instance sql-is-better:sql-is-better

Properties

Status:	Running
Version:	MySQL 5.5
Size:	D1
Pricing Plan:	Per use
Replication Type:	Synchronous
Disk Usage:	82 MB / 10 GB

Authorized Applications edit

sql-vs-nosql

Storage Usage (GB) | Month | Day | Hour | Minute

All times are UTC.

Google apis

SQL vs NoSQL

Dashboard | Logs | SQL Prompt | Backups

Instance settings | Actions | New instance...

- Overview
- Services
- Team
- API Access
- Billing
- Google Cloud SQL

SELECT ... FROM ... WHERE ...

Execute Database: performance_schema

Results

Execute an SQL statement to see results.

Google apis

SQL vs NoSQL

Dashboard | Logs | SQL Prompt | Backups

Instance settings | Actions | New instance...

- Overview
- Services
- Team
- API Access
- Billing
- Google Cloud SQL

```
CREATE DATABASE test;
```

Execute Database: performance_schema

Results

Execute an SQL statement to see results.

Google apis

SQL vs NoSQL

Dashboard | Logs | SQL Prompt | Backups

Instance settings | Actions | New instance...

- Overview
- Services
- Team
- API Access
- Billing
- Google Cloud SQL

```
CREATE DATABASE test;
```

Execute Database: performance_schema

Results 1 rows updated.

CREATE DATABASE test;	Jun 20, 2012 10:37 AM (224 ms)

Google apis

SQL vs NoSQL

Dashboard | Logs | SQL Prompt | Backups

Instance settings | Actions | New instance...

- Overview
- Services
- Team
- API Access
- Billing
- Google Cloud SQL

```
CREATE TABLE t1 (c1 INT, c2 VARCHAR(256));
```

Execute Database: performance_schema

Results 1 rows updated.

CREATE DATABASE test;	**Jun 20, 2012 10:37 AM (224 ms)**

Google apis

SQL vs NoSQL

Dashboard | Logs | SQL Prompt | Backups

Instance settings | Actions | New instance...

- Overview
- Services
- Team
- API Access
- Billing
- Google Cloud SQL

```
CREATE TABLE t1 (c1 INT, c2 VARCHAR(256));
```

Execute Database: performance_schema

Results 0 rows updated.

CREATE TABLE t1 (c1 INT, c2 VARCHAR(256));	**Jun 20, 2012 10:38 AM (474 ms)**

Using Datastore

- No configuration needed
- Just start writing data
- Entity ‘Kinds’ for table
- Namespaces for multi-tenancy/isolation

Scoreboard

	Datastore	Cloud SQL
Queries	✔	✔+
Transactions	✔	✔+
Consistency	✔	✔+
Scalability	✔+	✔
Management	✔+	✔

Schema

SQL Schema Change

- Strictly enforced
- Set at table create

Schema Change

SQL

```
CREATE TABLE Player (name VARCHAR(256), health int);
...
ALTER TABLE Player ADD COLUMN mana int;
```

SQL Schema Change

- ALTER TABLE
  - Locks the table
  - Copies entire table

- Online Schema Change
  - Write to new and old table
  - Bulk copy
  - Rename new table

  - Look at Percona's pt-online-schema-change for an example

Datastore Schema Change

- Update code
- Optionally write MapReduce to backfill

Python

```
class Player(db.Model)
  name = db.StringProperty()
  health = db.IntegerProperty()
  mana = db.IntegerProperty(default=0)
```

Scoreboard

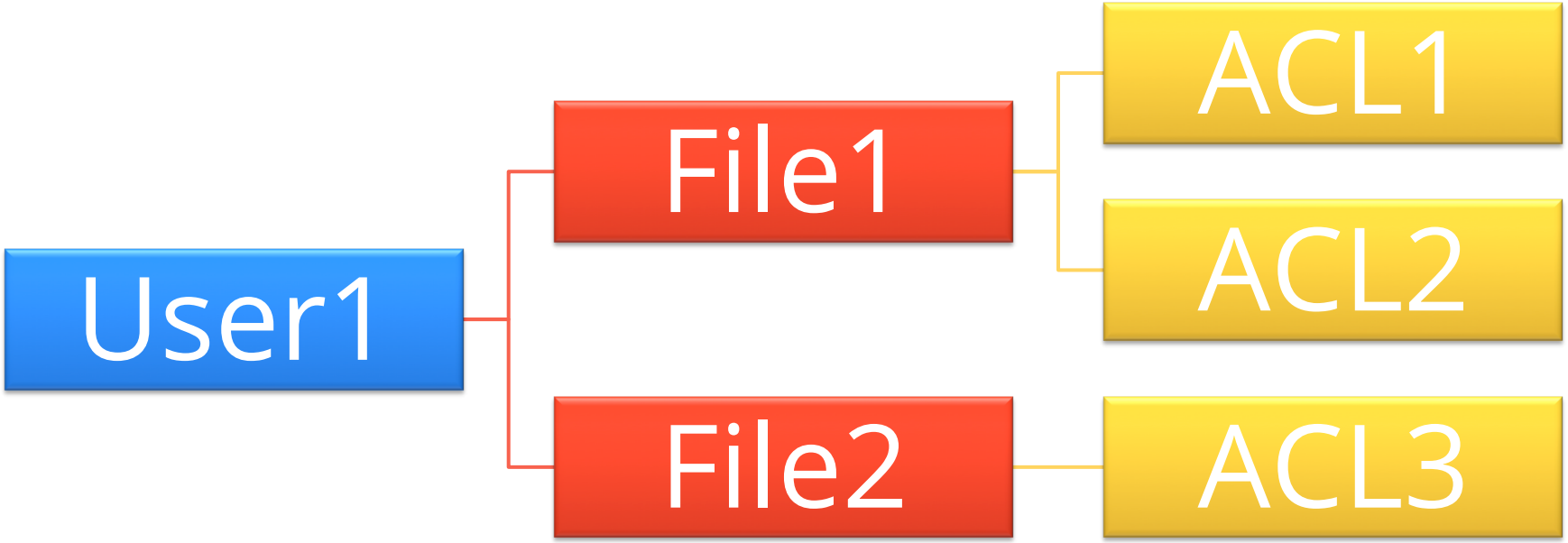
	Datastore	Cloud SQL
Queries	✔	✔+
Transactions	✔	✔+
Consistency	✔	✔+
Scalability	✔+	✔
Management	✔+	✔
Schema	✔+	✔

Friends?

DropRectangle.net SQL

Users	Files	ACL
• user_id • name	• file_id • owner_id • name	• file_id • user_id • permission

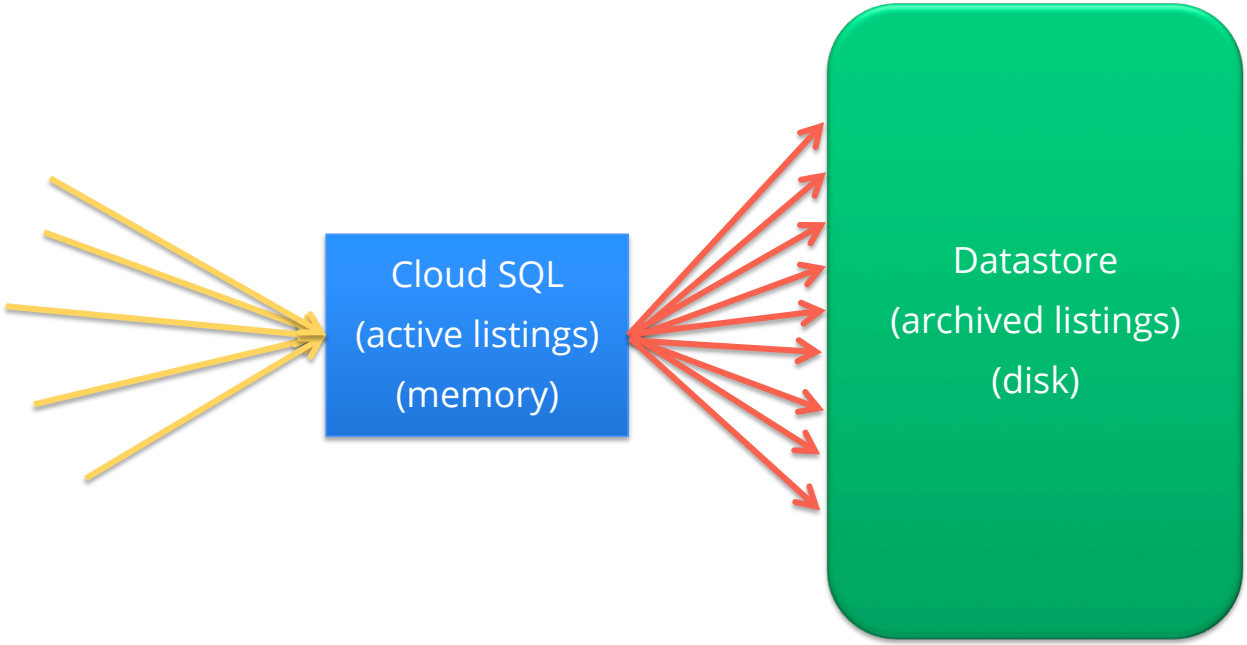
DropRectangle.net NoSQL



Full support of off-the-shelf

- Frameworks
  - Hibernate
  - JDO/JPA
  - Spring
  - Django
- WordPress
- Standards Based Existing Applications

Greg's List


Cloud SQL
(active listings)
(memory)
Datastore
(archived listings)
(disk)

Scoreboard

	Datastore	Cloud SQL
Queries	✔	✔+
Transactions	✔	✔+
Consistency	✔	✔+
Scalability	✔+	✔
Management	✔+	✔
Schema	✔+	✔

<Thank You!>

Questions?

https://developers.google.com/appengine/docs/python/datastore/
https://developers.google.com/cloud-sql/