

GCP Certification Series: 3.4

Deploying and implementing data solutions



Prashanta Paudel

Oct 31, 2018 · 22 min read

You may already know that Google is a data company rather than IT company because most of its product depends on analysis of data from its users.

For example, Google Search, the world's number one search engine depends on users statistics to show relevant search results and adds to the users. So, in short, it uses users data to show add to the same user. All Google does is processing the data.

If we look at the **data solutions from Google** cloud we can divide it into

Databases

- Cloud SQL
- Cloud Datastore
- Cloud Spanner
- Cloud Bigtable

There are other solutions also but we don't require that right now.

Data Analytics

- BigQuery
- Cloud Pub/Sub
- Cloud Dataproc

There are a lot more solutions but we don't need them now.

Databases

As we already studied before there are primarily two kinds of databases

1. SQL

- Cloud SQL
- Cloud Spanner

2. NoSQL

- Cloud Datastore
- Cloud Bigtable

Cloud SQL



Cloud SQL

Cloud SQL is a fully managed database service that makes it easy to set up, maintain, manage and administer your relational database. Cloud SQL supports

1. PostgreSQL and
2. MySQL

It offers high performance, scalable and convenience for users to create and implement a database for their applications. A database is structured data it will be hosted in Google cloud but can be accessed from anywhere.



Focus on Your Application

Let Google manage your database, so you can focus on your applications. Cloud SQL is perfect for Wordpress sites, e-commerce applications, CRM tools, geospatial applications, and any other application that is compatible with MySQL or PostgreSQL.

Simple & Fully Managed

Cloud SQL is easy to use. It doesn't require any software installation. It automates all your backups, replication, patches, and updates - while ensuring greater than 99.95% availability, anywhere in the world.



Performance & Scalability

Cloud SQL delivers high performance and scalability with up to 10TB of storage capacity, 40,000 IOPS, and 416GB of RAM per instance.



Reliability & Security

Easily configure replication and backups to protect your data. Go further by enabling automatic failover to make your database highly available (HA). Your data is automatically encrypted and Cloud SQL is SSAE 16, ISO 27001, PCI DSS v3.0, and HIPAA compliant.



Discounts Without Lock-in

Cloud SQL offers per-second billing, automatic sustained use discounts, and instance sizes to fit any budget. Database instances are easy to stop and start. There is no up-front commitment, and with sustained use discounts, you'll automatically get discounted prices for databases that run continuously.



References: <https://cloud.google.com/sql/>

CLOUD SQL FEATURES

Cloud SQL is a fully-managed MySQL and PostgreSQL database service.

Scalability

Easily scale up to 64 processor cores and more than 400GB of RAM. Quickly scale out with read replicas.

High Performance

Designed to scale from small development workloads up to performance-intensive workloads.

Integrated

Cloud SQL instances are accessible from just about any application, anywhere. Easily connect from [App Engine](#), [Compute Engine](#), and your workstation.

Fully Managed

Replicated, managed and backed-up, so you can make better use of your time.

Security

Cloud SQL data is encrypted when on Google's internal networks and when stored in database tables, temporary files, and backups. Cloud SQL supports private connectivity with [Virtual Private Cloud \(VPC\)](#) and every Cloud SQL instance includes a network firewall, allowing you to control public network access to your database instance. Learn more about Google Cloud Platform's [comprehensive security architecture](#).

Standard APIs

Build and deploy for the cloud faster because Cloud SQL offers standard MySQL and PostgreSQL databases. Use standard connection drivers and built-in migration tools to get started quickly.

Availability Protection

Live migration makes maintenance of our underlying infrastructure [transparent](#). For isolation from failures, [High Availability](#) provides continuous health-checking and automatically fails over if an instance is not healthy.

Partnerships & Integrations

Take advantage of our growing [partner ecosystem](#) and tools to make working with Cloud SQL even easier. Our partners can help you streamline the process of loading your data, create rich visualizations for meaningful insights, and monitor and manage your databases.

References: <https://cloud.google.com/sql/>

Cloud SQL Pricing depends on Regions and Zones of the GCP

Use cases

PRODUCT	DESCRIPTION	GOOD FOR	COMMON WORKLOADS
 Google Cloud SQL	A fully-managed MySQL and PostgreSQL database service that is built on the strength and reliability of Google's infrastructure.	<ul style="list-style-type: none">• Web frameworks• Structured data• OLTP workloads	<ul style="list-style-type: none">• Websites, blogs, and content management systems (CMS)• Business Intelligence (BI) applications• ERP, CRM, and eCommerce applications• Geospatial applications

Cloud SQL uses

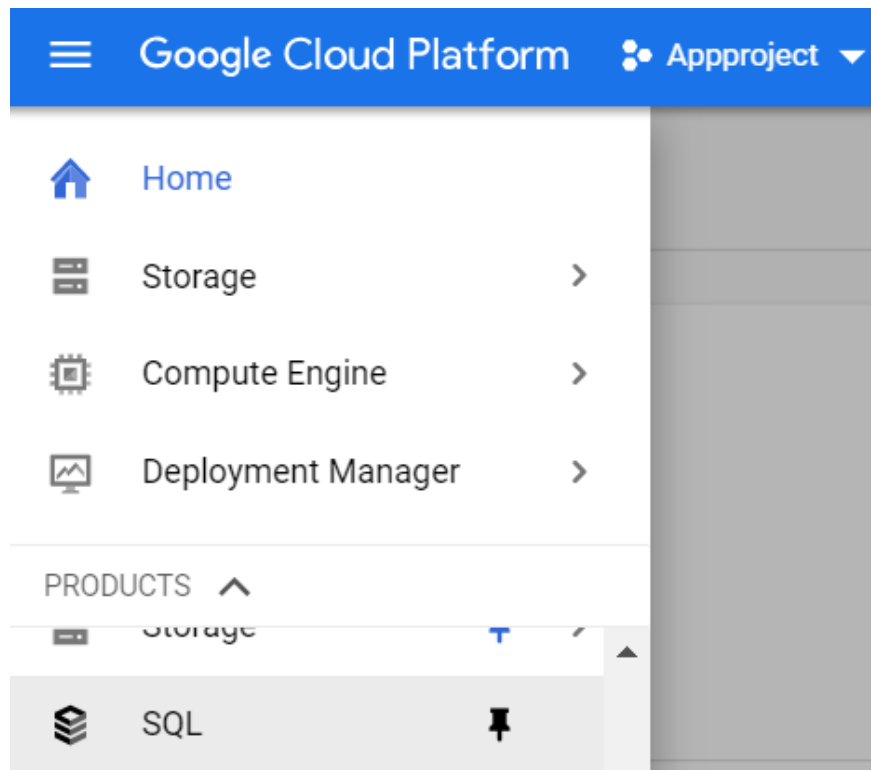
=====

=====

Example Project 1: Simplest MySQL connect

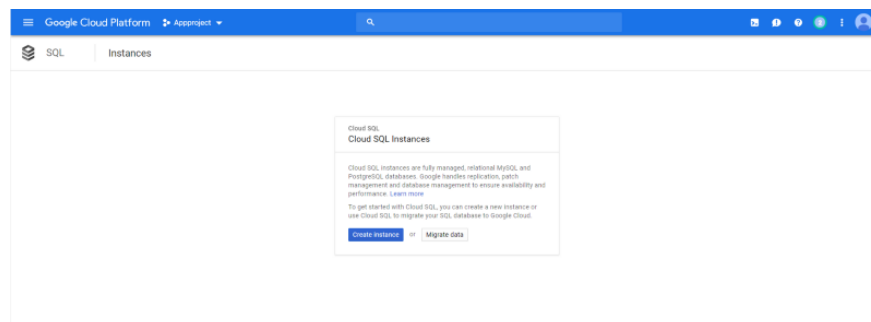
The simplest form of SQL connection is to create a SQL instance in Cloud and connect it from authorized user and IP to do some database tasks.

First of all, Create a database in Cloud SQL.



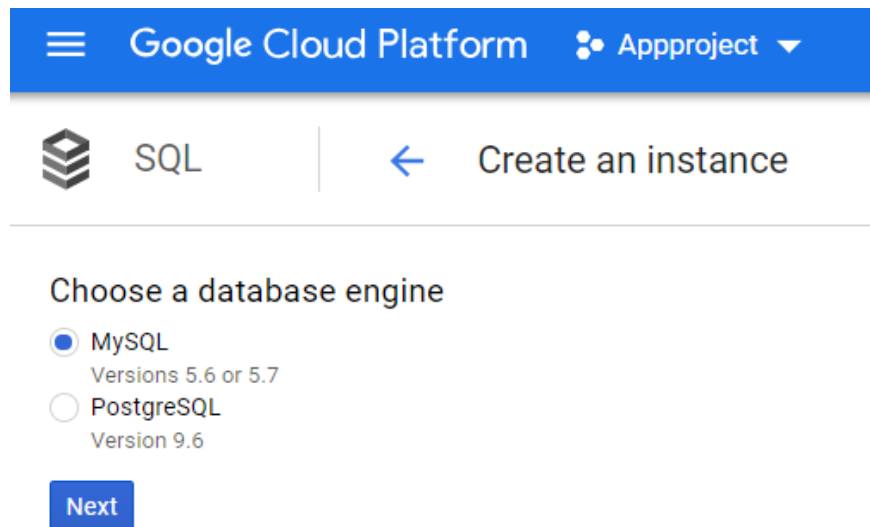
select SQL

Then you will see a Dashboard



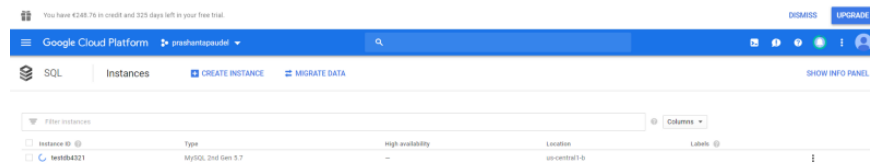
Cloud SQL Dashboard

Click on Create instance



Select MySQL

Now create the database



creating a database

After creating a database you should install Mysql Workbench in the client machine to be able to connect to MySQL server. For that download it in the Linux machine or any other if you wish.

```
$ sudo apt-get install mysql-workbench
```

This will install workbench in your Linux machine.

Now authenticate the connection by allowing the Linux IP in cloud SQL and add root password.

The screenshot shows the Google Cloud Platform console. At the top, there's a blue header bar with 'Edit network' and icons for delete and back. Below this is a form with two sections: 'Name (Optional)' with a text input containing 'vm', and 'Network' with a text input containing a blue bar and a link 'Use CIDR notation.' Below the form are 'Done' and 'Cancel' buttons. Underneath the form is a button with a plus icon and the text '+ Add network'. At the bottom of the dialog are 'Save' and 'Discard changes' buttons.

Below the dialog, there's a section for 'sample-demo' MySQL Second Generation master. It has tabs for 'OVERVIEW', 'CONNECTIONS', 'USERS', and 'DATABASES'. The 'USERS' tab is selected. The section title is 'MySQL user accounts'. Below the title is a description: 'User accounts enable users and applications to connect to your Cloud SQL instance. [Learn more](#)'. There's a blue button 'Create user account'. Below this is a table with two columns: 'User name' and 'Host name'. The table has two rows: 'mysql.sys' with 'localhost' and 'root' with '% (any host)'. To the right of each row is a vertical ellipsis menu. The menu for the 'root' row is open, showing 'Change password' and 'Delete' options.

User name	Host name
mysql.sys	localhost
root	% (any host)

Now goto Linux machine which has MySQL workbench and connects to the Cloud SQL Server we just created.

```
:~$ mysql --host=35.193.199.18 --user=root --password
```

Welcome to the MariaDB monitor. Commands end with; or \g. Your MySQL connection id is 57 Server version: 5.7.14-google-log (Google) Copyright © 2000, 2017, Oracle, MariaDB Corporation Ab, and others. Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

Now you are connected to Cloud SQL from CLI.

Hurray!

Some of the commands are

```
MySQL [(none)]> show databases
-> ;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
+-----+
4 rows in set (0.04 sec)

MySQL [(none)]> create database test
-> ;
Query OK, 1 row affected (0.04 sec)

MySQL [(none)]> create table people;
ERROR 1046 (3D000): No database selected
MySQL [(none)]> CREATE TABLE Persons (
->     PersonID int,
->     LastName varchar(255),
->     FirstName varchar(255),
->     Address varchar(255),
```

SQL commands

```
=====
=====
```


=====

=====


Example 2: Localhost DB-compute engine + LAMP(No cloud SQL)

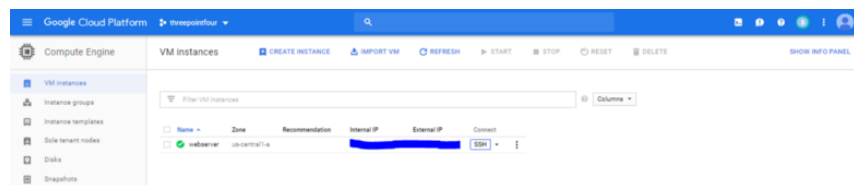
Guestbook website

First of all, let's create a web server, please follow the steps in the blogs earlier to set up a web server

Google Cloud Platform(GCP): Building a web server to change PHP form input into a...

In this example, I am going to show how to build a simple GCP based application that will take input... medium.com





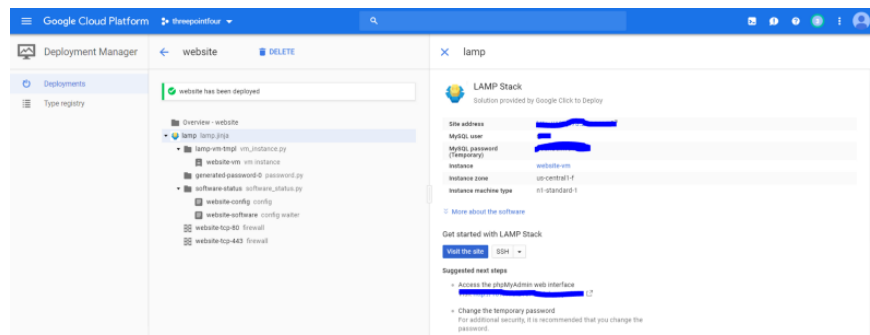
New web server created

Here, I created a web server with Debian OS and install

- Apache2
- MySQL Workbench
- PHP
- phpmyadmin

In short, it is called a LAMP server.

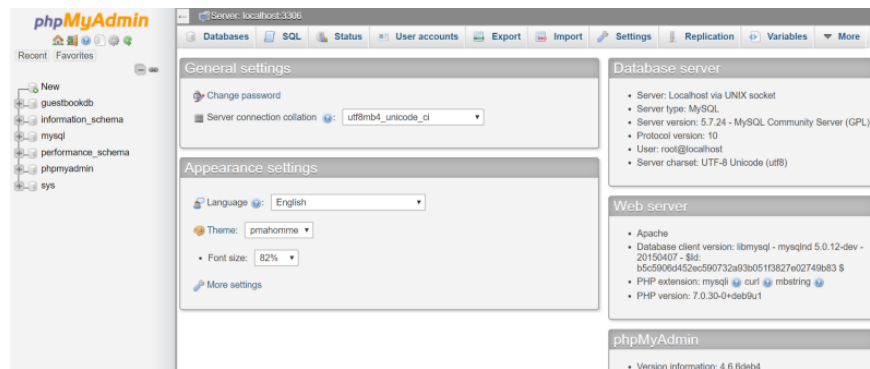
So, instead of installing each package one by one I deployed a LAMP server from Marketplace in Google cloud.



LAMP Server

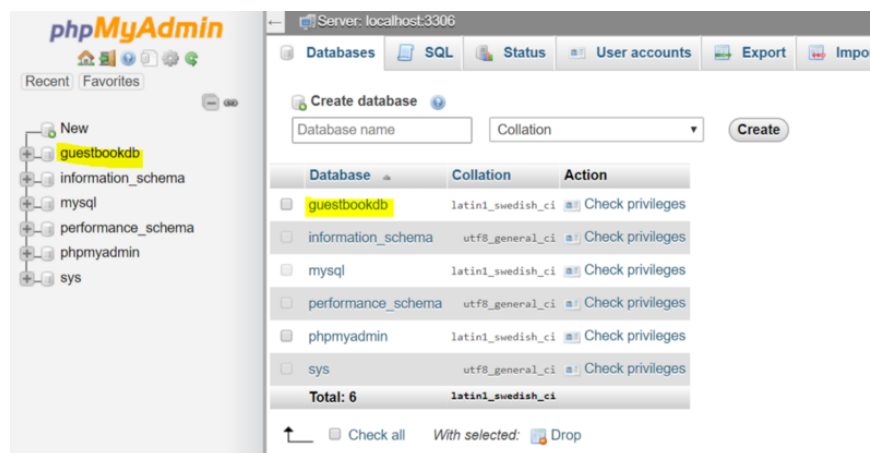
Now that we have all the infrastructure, we will start building the database first.

Goto <http://website/phpmyadmin>



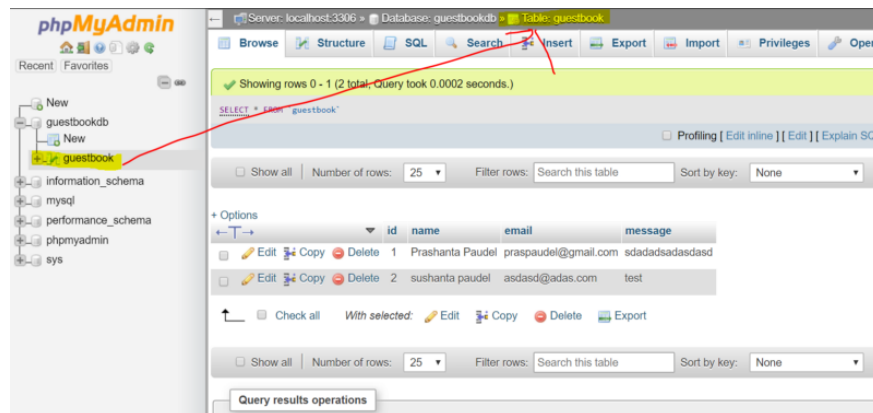
phpmyadmin page

Now create a database and table



Create a database

Now create table



Table

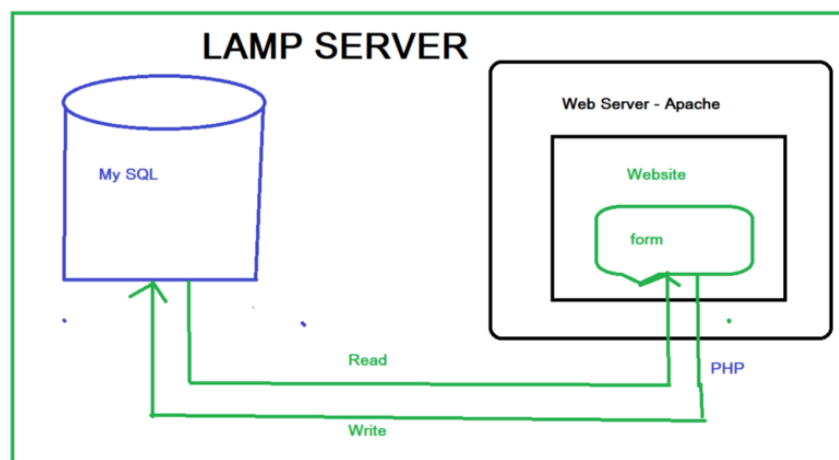
Now let's go to the web part,

```
root@website-vm:/var/www/html# ls
addcomment.php  guestbook.php  index.html
```

web server

We need three files to be hosted in the Web server

1. index.html for the front page of the website.
2. guestbook.php shows all the list of signed guests.
3. addcomment.php to add web front end form data to the database.



Lamp server

index.html

```

<head>
<script type="text/javascript"> // <![CDATA[
function Validate()
{
var x=document.forms["guest"]["email"].value;
var y=document.forms["guest"]["name"].value;
if(y==null || y=="")
{
alert("Please enter your Name! ");
return false;
}
if(x==null || x=="")
{
alert("Please enter your email address!");
return false;
}
var atpos=x.indexOf("@");
var dotpos=x.lastIndexOf(".");
if (atpos<1 || dotpos<atpos+2 || "" dotpos+2!=">=x.length)
{
alert("Not a valid e-mail address");
return false;
}
else
return true;
}
</atpos+2>
// ]]></script>
</head>

<body>
<a href="guestbook.php">Display All</a>
</pre>
<form action="addcomment.php" method="post" name="guest" onsubmit="return Validate();" >
Name: <input type="text" name="name" />
Email: <input type="text" name="email" />
Message:

<textarea cols="50" name="message" rows="10"> </textarea>
<input type="submit" value="Sign this in the Book" /></form>
<pre>
</body>

```

guestbook.php

```

<?php
$host="localhost"; //Add your SQL Server host here
$user="root"; //SQL Username
$pass="pacific"; //SQL Password
$dbname="guestbookdb"; //SQL Database Name
$con=mysqli_connect($host,$user,$pass,$dbname);
if (mysqli_connect_errno($con))
{
echo "Failed to connect to MySQL: " . mysqli_connect_error();
}
$result = mysqli_query($con,"SELECT name,message FROM guestbook");
while($row = mysqli_fetch_array($result))
{
<h3><?php echo $row['name']; ?></h3>
<p><?php echo $row['message']; ?></p>
<?php }
mysqli_close($con);
?>

```

addcomment.php

```

?php
host="localhost"; //Add your SQL Server host here
user="root"; //SQL Username
pass="password"; //SQL Password
dbname="guestbookdb"; //SQL Database Name
con=mysqli_connect($host,$user,$pass,$dbname);
if (mysqli_connect_errno($con))

echo "<h1>Failed to connect to MySQL: " . mysqli_connect_error() . "</h1>";

name=$_POST['name'];
email=$_POST['email'];
message=$_POST['message'];
sql="INSERT INTO guestbook(name,email,message) VALUES('$name','$email','$message')";
if (!mysqli_query($con,$sql))

die('Error: ' . mysqli_error($con));

else
echo "Values Stored in our Database!";
mysqli_close($con);

```

Front page

[Display ALL](#)

Name: Email: Message:

front page

=====

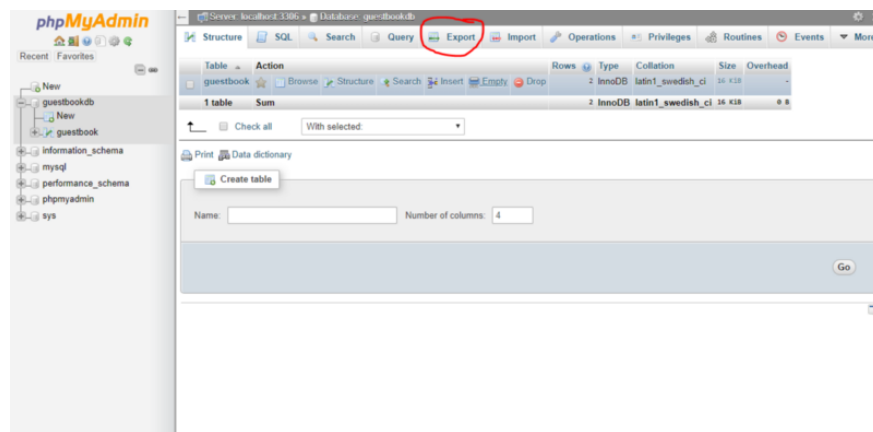
=====

Example 3: Cloud SQL and Compute Engine(PHP based)

Using web page to input and retrieve data to Cloud SQL

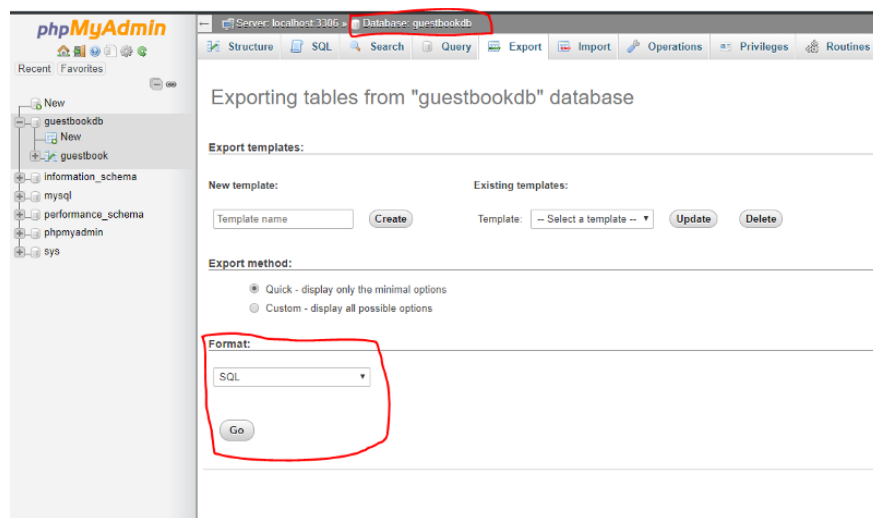
Now let's move the localhost database to the cloud using PHPMyAdmin

goto phpmyadmin



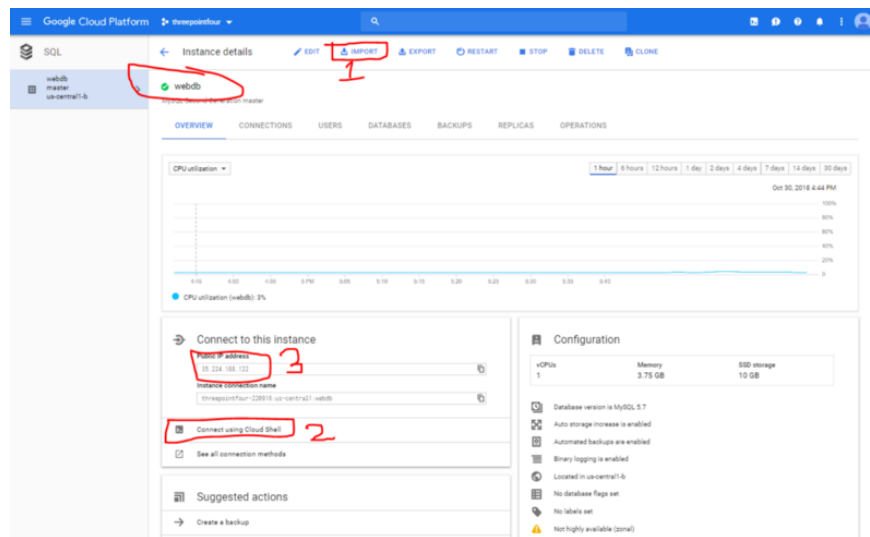
Export DB

Select the database and click export, in format select SQL and save the file into your desktop

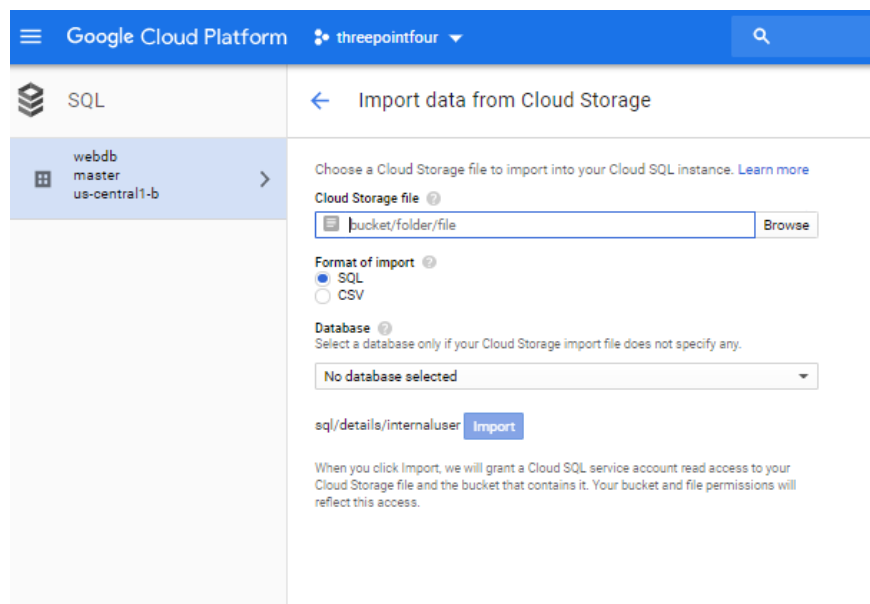


export db

Now Setup Cloud SQL in GCP



After starting the Cloud SQL server, the first thing to do is import the database we exported from phpmyadmin. This way we will get data from earlier.



One thing to remember is, we cannot import directly but have to upload into a bucket in Cloud Storage and then select from there.

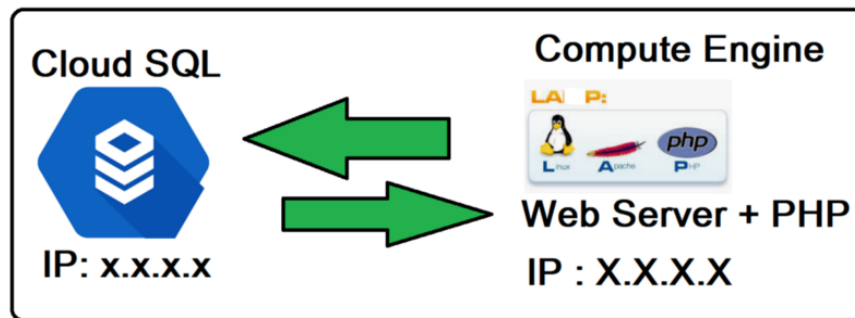
after import check the database, select database and see tables and data.

Now we have to edit the addcomment.php and guestbook.php file. Basically, we have to redirect database server address to Cloud SQL and

that is 4 lines in php database connect lines

```
<?php
$host="10.10.10.10" //Add your SQL Server host here
$user="root"; //SQL Username
$password="root"; //SQL Password
$dbname="guestbookdb"; //SQL Database Name
$con=mysqli_connect($host,$user,$password,$dbname);
if (mysqli_connect_errno($con))
{
    echo "Failed to connect to MySQL: " . mysqli_connect_error();
}
$result = mysqli_query($con,"SELECT name,message FROM guestbook");
while($row = mysqli_fetch_array($result))
{
    ?>
    <h3><?php echo $row['name']; ?></h3>
    <p><?php echo $row['message']; ?></p>
    <?php }
    mysqli_close($con);
?>
```

So, Now the Architecture of our system is



Cloud SQL + Compute

=====

=====

LOADING DATA TO Cloud SQL

1. Import/ Export: In the example
2. Load Data from Cloud Storage: buckets used in transfer.

=====

=====

Cloud Datastore




Cloud Datastore

Cloud Datastore is a highly scalable NoSQL database for your web and mobile applications.

Google Cloud Datastore gives you an elastic, highly available document-oriented database as a service. It is fully managed so you don't need to deploy, update, configure, or manage your database solution. Cloud Datastore comes with a rich admin dashboard, a powerful query engine as well as multiple methods to access the database which makes it ideal for mobile and web workloads.

Highly Scalable NoSQL Database

Cloud Datastore is a highly-scalable NoSQL database for your applications. Cloud Datastore **automatically handles sharding and replication**, providing you with a highly available and durable database that scales automatically to handle your applications' load. Cloud Datastore provides a myriad of capabilities such as **ACID transactions, SQL-like queries, indexes and much more.**




Simple & Integrated

With Cloud Datastore's **RESTful interface**, data can easily be accessed by any deployment target. You can build solutions that span across **App Engine and Compute Engine**, and rely on Cloud Datastore as the integration point.

Fast & Highly Scalable

Focus on building your applications without worrying about provisioning and load anticipation. Cloud Datastore **scales seamlessly and automatically** with your data allowing applications to maintain high performance as they receive more traffic.



Easy to Use Query Language

Datastore is a schemaless database, which allows you to worry less about making changes to your underlying data structure as your application evolves. Datastore **provides a powerful query engine** that allows you to search for data across multiple properties and sort as needed.

```

1. // List Google companies with fewer than 400 employees.
2. var companies = query.filter('name =', 'Google').filter('size <', 400);
3.

```

References: <https://cloud.google.com/datastore/>

CLOUD DATASTORE FEATURES

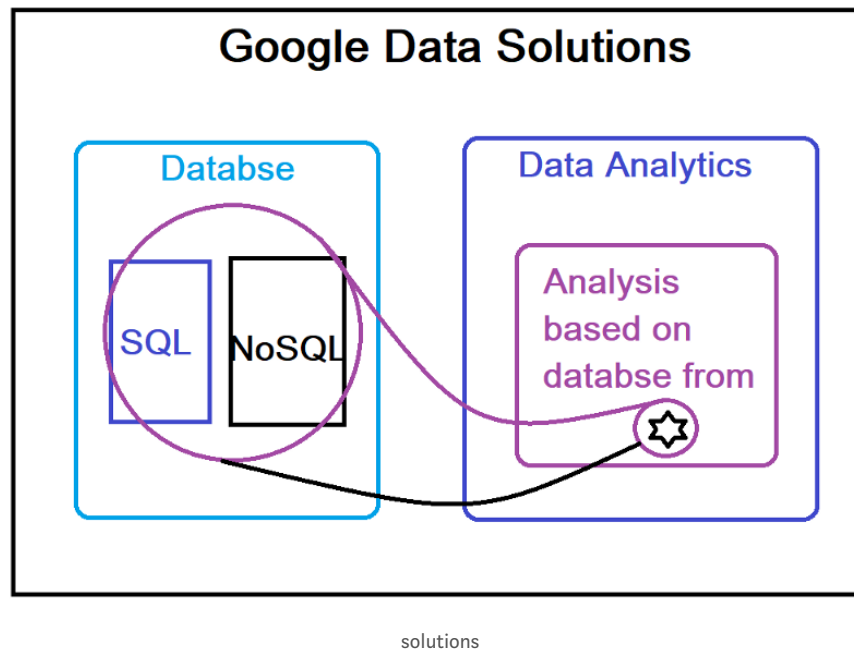
Cloud Datastore is a highly-scalable NoSQL database for your web and mobile applications

<h3>Rich Admin Dashboard</h3> <p>View entity statistics, query your database, view indexes, and backup/restore your data.</p>	<h3>Diverse Data Types</h3> <p>Datastore supports a variety of data types, including integers, floating-point numbers, strings, dates, and binary data among others.</p>
<h3>Multiple Access Methods</h3> <p>Access your data via our JSON API, open-source clients, or community maintained ORMs (Objectify, NDB).</p>	<h3>ACID Transactions</h3> <p>Ensure the integrity of your data by executing multiple datastore operations in a single transaction with ACID characteristics, so all the grouped operations succeed or all fail.</p>
<h3>Fully Managed</h3> <p>Cloud Datastore is fully managed, which means Google automatically handles sharding and replication in order to provide you with a highly available and consistent database.</p>	

references: <https://cloud.google.com/datastore/>

After reading the intro one thing that comes to every mind is how to use it. Since it is Database storing platform for NoSQL databases, the application will store databases to it not files or folders.

Let me clear the difference here by

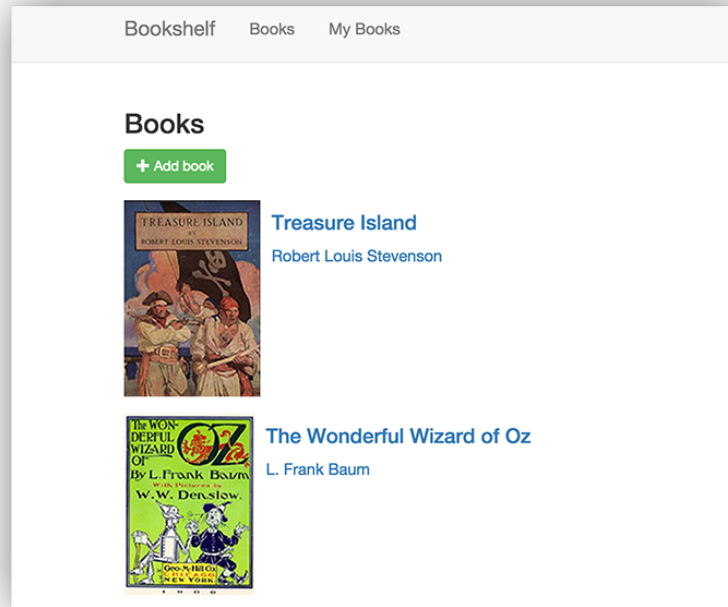


So, let's do one example.

PHP Bookshelf App

The Bookshelf app is a sample web app written in PHP that shows how to use a variety of Google Cloud Platform products, including:

- Google App Engine flexible environment
- Google Cloud SQL
- Google Cloud Datastore
- Google Cloud Storage
- Google Cloud Pub/Sub



The Bookshelf sample app stores a collection of book titles. Anyone who has access to the app can add books to the list. The sample app offers these features:

- Users can view the list of books, add books to the list, and remove books from the list.
- Users can edit book details.
- Users can upload cover images for books.
- Users can log in with their Google accounts and view the books that they have added to the list.

. . .

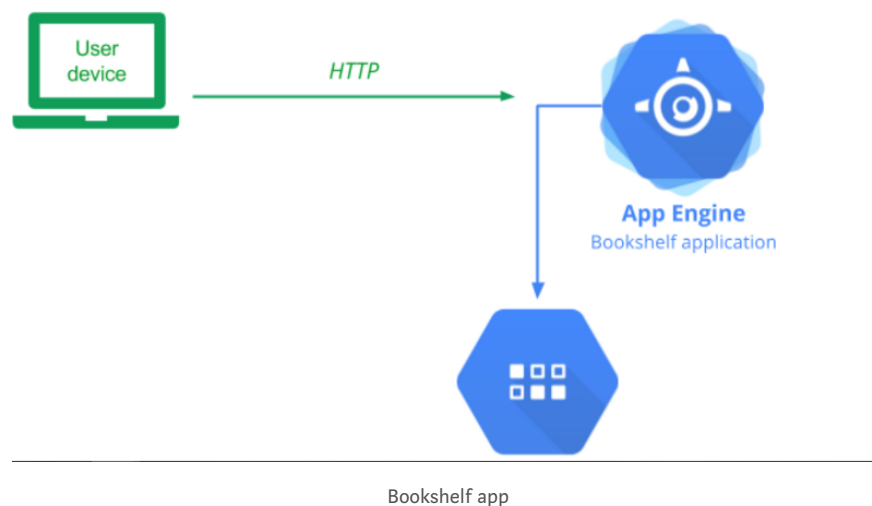
Objectives

- Clone or download the sample app.
- Build the app and run it on your local machine.
- Deploy the app to App Engine.
- Walk through the sample code.

- Learn how the app stores structured data.
- Learn how the app stores binary data in Google Cloud Storage.
- Learn how the app authenticates users.
- Learn how the app creates event logs that are visible in the Google Cloud Platform Console.

This tutorial explores the Bookshelf app in detail and discusses how each feature of the app is implemented using familiar technologies and services provided by Cloud Platform.

Build a bookshelf application that runs in app engine but saves the database in Cloud Datastore.

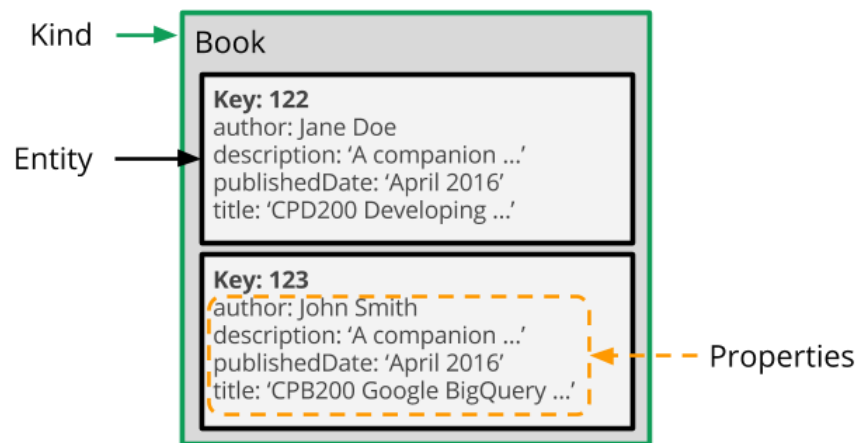


A kind in Cloud Datastore is roughly equivalent to a table in a relational database. Kinds are used to categorize entities for queries.

Data objects in Google Cloud Datastore are known as entities. An entity has one or more named properties (which are like fields in a relational database). Each property can have one or more values. Entities are identified by a unique key and are akin to rows in a relational database. Entities of the same kind need not have the same properties. These unique characteristics imply a different way of designing and managing data to take advantage of the ability to scale automatically.

Cloud Datastore also organizes data into entity groups. An entity group consists of a root entity and all of its descendants. Applications typically use entity groups to organize highly related data to achieve strong consistency and to enforce transactionality. For example, an application could use an entity group to store data about one product, or one user profile.

The Bookshelf data model for Cloud Datastore is illustrated by the following diagram. Note that the simple data model used in Bookshelf does not make use of entity groups.



Before you begin

- Create a new GCP project, and then create an App Engine application and enable billing in that project.
- Enable the Cloud Datastore, Cloud Pub/Sub, Cloud Storage JSON, Stackdriver Logging, Google+, and Google Cloud SQL APIs.

Verify that your default project is correct:

```
gcloud config list
```

If the project ID listed in the output is not the project that you intended to use for this tutorial, set the project by entering this command:

```
gcloud config set project [YOUR_PROJECT_ID]
```

where `[YOUR_PROJECT_ID]` is the ID of the project you created or chose to use for this tutorial.

Clone the sample repository:

```
git clone https://github.com/GoogleCloudPlatform/getting-started-php.git
```

Alternatively, you can [download the sample](#) as a zip file and extract it.

Now, here comes main steps:

Step 1: Clone repository from github

```
prashantagcppaudel@cloudshell:~ (pfour-221109)$ git clone https://github.com/GoogleCloudPlatform/getting-started-php.git
Cloning into 'getting-started-php'...
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 2143 (delta 0), reused 2 (delta 0), pack-reused 2136
Receiving objects: 100% (2143/2143), 1.04 MiB | 0 bytes/s, done.
Resolving deltas: 100% (1419/1419), done.
```

Step 2: Goto getting started folder and 2-structured-data folder

```
$ cd getting-started-php/
$ cd 2-structured-data/
```

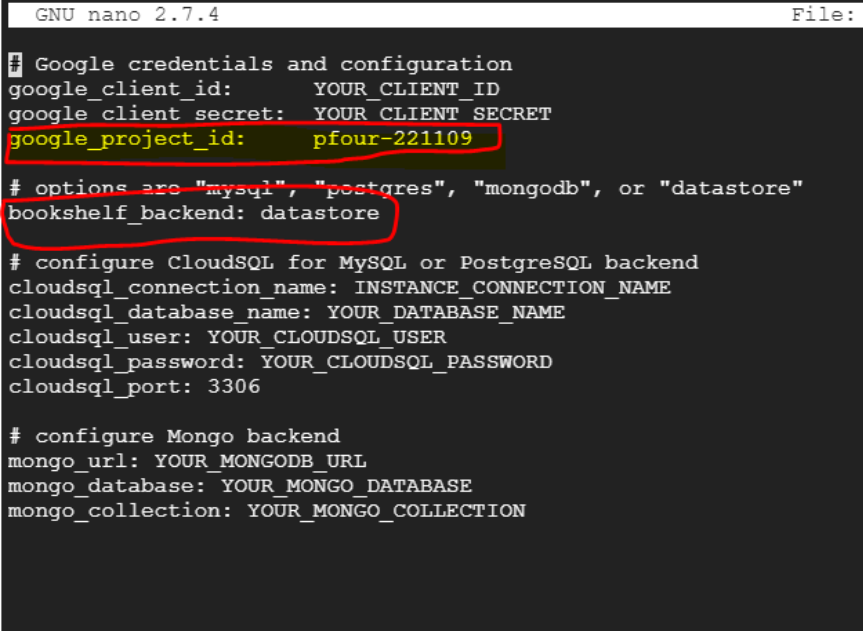
Step 3: copy settings.yaml.dist into settings.yaml

```
$ cp config/settings.yml.dist config/settings.yml
```

Step 4: Again goto config folder

```
$ cd config/
```

Step 5: Edit settings .yml as follows



```
GNU nano 2.7.4 File: settings.yml
# Google credentials and configuration
google_client_id: YOUR_CLIENT_ID
google_client_secret: YOUR_CLIENT_SECRET
google_project_id: pfour-221109
# options are "mysql", "postgres", "mongodb", or "datastore"
bookshelf_backend: datastore
# configure CloudSQL for MySQL or PostgreSQL backend
cloudsql_connection_name: INSTANCE_CONNECTION_NAME
cloudsql_database_name: YOUR_DATABASE_NAME
cloudsql_user: YOUR_CLOUDSQL_USER
cloudsql_password: YOUR_CLOUDSQL_PASSWORD
cloudsql_port: 3306
# configure Mongo backend
mongo_url: YOUR_MONGODB_URL
mongo_database: YOUR_MONGO_DATABASE
mongo_collection: YOUR_MONGO_COLLECTION
```

Step 6: Install composer

```
$ composer install
Loading composer repositories with package information
Installing dependencies (including require-dev) from lock
file
Package operations: 52 installs, 0 updates, 0 removals
- Installing firebase/php-jwt (v4.0.0): Downloading (100%)
- Installing rize/uri-template (0.3.2): Downloading (100%)
- Installing psr/http-message (1.0.1): Downloading (100%)
- Installing psr/log (1.0.2): Downloading (100%)
- Installing monolog/monolog (1.23.0): Downloading (100%)
- Installing guzzlehttp/psr7 (1.4.2): Downloading (100%)
- Installing guzzlehttp/promises (v1.3.1): Downloading
(100%)
```



```

- Installing guzzlehttp/guzzle (6.3.0): Downloading (100%)
- Installing psr/cache (1.0.1): Downloading (100%)
- Installing google/auth (v0.11.1): Downloading (100%)
- Installing google/cloud (v0.21.1): Downloading (100%)
- Installing symfony/routing (v3.0.9): Downloading (100%)
- Installing symfony/polyfill-mbstring (v1.7.0):
Downloading (100%)
- Installing symfony/http-foundation (v3.0.9): Downloading
(100%)
- Installing symfony/event-dispatcher (v3.0.9):
Downloading (100%)
- Installing symfony/debug (v3.4.4): Downloading (100%)
- Installing symfony/http-kernel (v3.0.9): Downloading
(100%)
- Installing pimple/pimple (v1.1.1): Downloading (100%)
- Installing silex/silex (v1.3.6): Downloading
(100%)monolog/monolog suggests installing php-amqplib/php-
amqplib (Allow sending log messages to an AMQP server using
php-amqplib)
monolog/monolog suggests installing php-console/php-console
(Allow sending log messages to Google Chrome)
monolog/monolog suggests installing rollbar/rollbar (Allow
sending log messages to Rollbar)
monolog/monolog suggests installing ruflin/elastica (Allow
sending log messages to an Elastic Search server)
monolog/monolog suggests installing sentry/sentry (Allow
sending log messages to a Sentry server)
google/cloud suggests installing google/gax (Required to
support gRPC)
google/cloud suggests installing google/proto-client-php
(Required to support gRPC)
google/cloud suggests installing james-heinrich/getid3
(Allows the Google Cloud Speech client to determine sample
rate and encoding of audio inputs)
symfony/routing suggests installing doctrine/annotations
(For using the annotation loader)
symfony/routing suggests installing symfony/config (For
using the all-in-one router or any loader)
symfony/routing suggests installing symfony/dependency-
injection (For loading routes from a service)
symfony/routing suggests installing symfony/expression-
language (For using expression matching)
symfony/event-dispatcher suggests installing
symfony/dependency-injection
symfony/http-kernel suggests installing symfony/class-loader
symfony/http-kernel suggests installing symfony/config
symfony/http-kernel suggests installing symfony/dependency-
injection
symfony/http-kernel suggests installing symfony/finder
symfony/http-kernel suggests installing symfony/var-dumper
behat/mink suggests installing behat/mink-selenium2-driver
(slow, but JS-enabled driver for any app (requires
Selenium2))
behat/mink suggests installing behat/mink-zombie-driver
(fast and JS-enabled headless driver for any app (requires
node.js))
symfony/console suggests installing symfony/lock
sebastian/global-state suggests installing ext-uopz (*)
phpunit/phpunit-mock-objects suggests installing ext-soap
(*)

```

```
phpunit/php-code-coverage suggests installing ext-xdebug
(>=2.2.1)
phpunit/phpunit suggests installing phpunit/php-invoker
(~1.1)
Generating autoload files
```

Step 7: Deploy App in App Engine

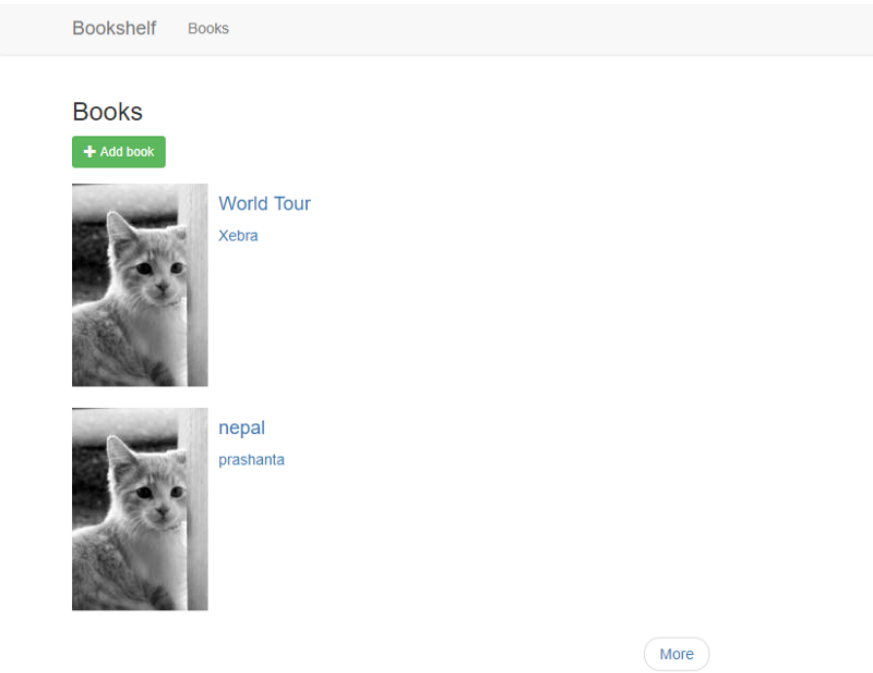
```
$ gcloud app deploy
3acd409e5aea: Layer already exists
363b7c2debba: Pushed
ed59ef025e1e: Layer already exists
7420484a2283: Pushed
5bdaf3588472: Layer already exists
03fce3219a27: Layer already exists
56f56dd1c2db: Layer already exists
2f4254df2a74: Layer already exists
3bfe3bd8f9e5: Pushed
89d0200fdea0: Layer already exists
f7fb3b0f4fc6: Layer already exists
d464d4f089a3: Layer already exists
62f57a66e54a: Layer already exists
1c1171b5f438: Layer already exists
b00f214424ac: Layer already exists
96e208368397: Layer already exists
8c468b8e19fa: Layer already exists
b056439591da: Layer already exists
84ff92691f90: Layer already exists
latest: digest:
sha256:401c366752609e012efa055233becd40dce65dcb6ad70a23e820a
2370fd5f50f size: 4911
DONE
-----
-----
-----

Updating service [default] (this may take several
minutes)...done.
Setting traffic split for service [default]...done.
Deployed service [default] to [https://pfour-
221109.appspot.com]

You can stream logs from the command line by running:
$ gcloud app logs tail -s default

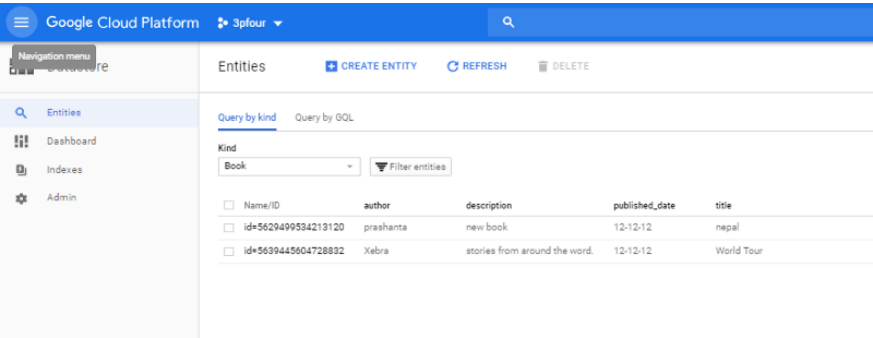
To view your application in the web browser run:
$ gcloud app browse
```

Step 8: Now check the website on <https://pfour-221109.appspot.com>.



FINISHED !

The information is stored in datastore as folows



=====

=====

Google BigQuery



Big Query

BigQuery is a fast, highly scalable, cost-effective, and fully managed cloud data warehouse for analytics, with built-in machine learning.

BigQuery is Google's serverless, highly scalable, enterprise data warehouse designed to make all your data analysts productive at an unmatched price-performance. Because there is no infrastructure to manage, you can focus on analyzing data to find meaningful insights using familiar SQL without the need for a database administrator.

Analyze all your data by creating a logical data warehouse over managed, columnar storage, as well as data from object storage and spreadsheets. Build and operationalize machine learning solutions with simple SQL. Easily and securely share insights within your organization and beyond as datasets, queries, spreadsheets, and reports.

BigQuery allows organizations to capture and analyze data in real time using its powerful streaming ingestion capability so that your insights are always current, and it's **free for up to 1 TB of data analyzed each month and 10 GB of data stored.**

This embedded content is from a site that does not comply with the Do Not Track (DNT) setting now enabled on your browser.

Please note, if you click through and view it anyway, you may be tracked by the website hosting the embed.

Learn More about Medium's DNT policy

This embedded content is from a site that does not comply with the Do Not Track (DNT) setting now enabled on your browser.

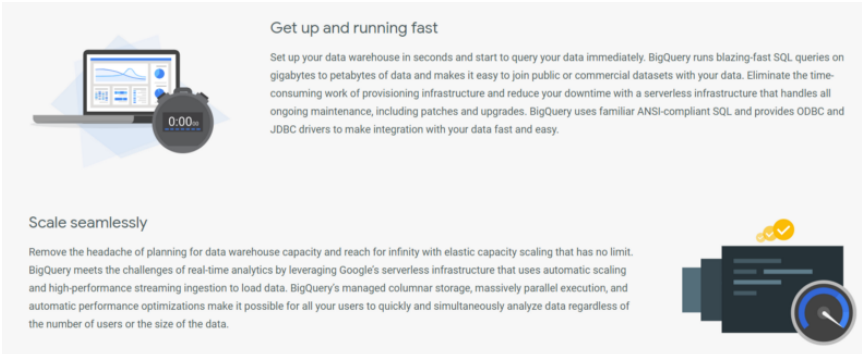
Please note, if you click through and view it anyway, you may be tracked by the website hosting the embed.

Learn More about Medium's DNT policy

This embedded content is from a site that does not comply with the Do Not Track (DNT) setting now enabled on your browser.

Please note, if you click through and view it anyway, you may be tracked by the website hosting the embed.

Learn More about Medium's DNT policy




Get up and running fast

Set up your data warehouse in seconds and start to query your data immediately. BigQuery runs blazing-fast SQL queries on gigabytes to petabytes of data and makes it easy to join public or commercial datasets with your data. Eliminate the time-consuming work of provisioning infrastructure and reduce your downtime with a serverless infrastructure that handles all ongoing maintenance, including patches and upgrades. BigQuery uses familiar ANSI-compliant SQL and provides ODBC and JDBC drivers to make integration with your data fast and easy.

Scale seamlessly

Remove the headache of planning for data warehouse capacity and reach for infinity with elastic capacity scaling that has no limit. BigQuery meets the challenges of real-time analytics by leveraging Google's serverless infrastructure that uses automatic scaling and high-performance streaming ingestion to load data. BigQuery's managed columnar storage, massively parallel execution, and automatic performance optimizations make it possible for all your users to quickly and simultaneously analyze data regardless of the number of users or the size of the data.

Reference: <https://cloud.google.com/bigquery/>




Accelerate your insights with powerful analysis

Get insights from your data faster without needing to copy or move it. Google BigQuery gives you full view of all your data by seamlessly querying data stored in BigQuery's managed columnar storage, Cloud Storage, Cloud Bigtable, Sheets, and Drive. BigQuery integrates with existing ETL tools like Informatica and Talend to enrich the data you already use. BigQuery supports popular BI tools like Tableau, MicroStrategy, Looker, and Data Studio out of the box, so anyone can easily create stunning reports and dashboards. Automatically ingest and visualize Google Ads and marketing data using BigQuery Data Transfer Service to set up a high-powered marketing data warehouse in just a few clicks.

Protect your business data and investments

Experience unmatched performance, security, and functionality for a cost that fits your budget. BigQuery eliminates the data operations burden by providing automatic data replication for disaster recovery and high availability of processing for no additional charge. BigQuery offers a 99.9% SLA and adheres to the Privacy Shield Principles. BigQuery makes it easy to maintain strong security with fine-grained identity and access-management control. BigQuery data is always encrypted, at rest and in transit.



Reference: <https://cloud.google.com/bigquery/>

BIGQUERY FEATURES

A fast, highly scalable, cost-effective, and fully managed enterprise data warehouse for analytics.

Serverless

Serverless data warehousing gives you the resources you need, when you need them. With BigQuery, you can focus on your data and analysis, rather than operating and sizing computing resources.

Real-time Analytics

BigQuery's high-speed streaming insertion API provides a powerful foundation for real-time analytics. BigQuery allows you to analyze what's happening now by making your latest business data immediately available for analysis.

Automatic High Availability

Free data and compute replication in multiple locations means your data is available for query even in the case of extreme failure modes. BigQuery transparently and automatically provides durable, replicated storage and high availability with no extra charge and no additional setup.

Standard SQL

BigQuery supports a standard SQL dialect which is ANSI:2011 compliant, reducing the need for code rewrite and allowing you to take advantage of advanced SQL features. BigQuery provides free ODBC and JDBC drivers to ensure your current applications can interact with BigQuery's powerful engine.

Petabyte Scale

BigQuery is fast and easy to use on data of any size. With BigQuery, you'll get great performance on your data, while knowing you can scale seamlessly to store and analyze petabytes more without having to buy more capacity.

Flexible Pricing Models

BigQuery enables you to choose the pricing model that best suits you. On-demand pricing lets you pay only for the storage and compute that you use. Flat-rate pricing enables high-volume users or enterprises to choose a stable monthly cost for analysis. For more information see [BigQuery pricing](#).

Data Encryption and Security

You have full control over who has access to the data stored in BigQuery. BigQuery makes it easy to maintain strong security with fine-grained identity and access management with [Cloud Identity and Access Management](#), and your data is always encrypted at rest and in transit.

Data Locality

You have the option to store your BigQuery data in US, Japan, and European locations while continuing to benefit from a fully managed service. BigQuery gives you the option of geographic data control, without the headaches of setting up and managing clusters and other computing resources in-region.

Reference: <https://cloud.google.com/bigquery/>

Standard SQL

BigQuery supports a standard SQL dialect which is ANSI:2011 compliant, reducing the need for code rewrite and allowing you to take advantage of advanced SQL features. BigQuery provides free ODBC and JDBC drivers to ensure your current applications can interact with BigQuery's powerful engine.

Federated Query and Logical Data Warehousing

BigQuery breaks down data silos so you can analyze all your data assets from one place. Through powerful federated query, BigQuery can process data in object storage (Cloud Storage), transactional databases (Cloud Bigtable), or spreadsheets in Google Drive — all without duplicating data. One tool lets you query all your data sources.

Storage and Compute Separation

BigQuery provides you with fine-grained control of cost and access. With BigQuery's separated storage and compute, you pay only for the resources you use. You have the option to choose the storage and processing solutions that make sense for your business and control access for each.

Automatic Backup and Easy Restore

BigQuery automatically replicates data and keeps a seven-day history of changes, reducing worries about unexpected data changes. This allows you to easily restore and compare data from different times.

Data Locality

You have the option to store your BigQuery data in US, Japan, and European locations while continuing to benefit from a fully managed service. BigQuery gives you the option of geographic data control, without the headaches of setting up and managing clusters and other computing resources in-region.

Foundation for AI

BigQuery provides a flexible, powerful foundation for machine learning and artificial intelligence. Besides bringing ML to your data with BigQuery ML, integrations with Cloud ML Engine and TensorFlow enable you to train powerful models on structured data. Moreover, BigQuery's ability to transform and analyze data helps you get your data in shape for machine learning.

Foundation for BI

BigQuery forms the data warehousing backbone for [modern BI solutions](#), and enables seamless data integration, transformation, analysis, visualization, and reporting with tools from Google and our technology partners.

Flexible Data Ingestion

Load your data from Cloud Storage or stream it into BigQuery at thousands of rows per second to enable real-time analysis of your data. Use familiar data integration tools like Informatica, Talend, and others out of the box.

Reference: <https://cloud.google.com/bigquery/>

Geospatial Datatypes and Functions

BigQuery GIS^{BETA} brings SQL support for the most commonly used GIS functions right into your data warehouse. With support for arbitrary points, lines, polygons, and multi-polygons in WKT and GeoJSON format, you can simplify your geospatial analyses, see your location-based data in new ways, or unlock entirely new lines of business with the power of BigQuery.

Data Transfer Service

BigQuery makes it easy to get started with data warehousing, even if your data is in a SaaS application. The [BigQuery Data Transfer Service](#) automatically transfers data from external data sources, like Google Marketing Platform, Google Ads, and YouTube, to BigQuery on a scheduled and fully managed basis.

Big Data Ecosystem Integration

With Cloud Dataproc and Cloud Dataflow, BigQuery provides integration with the Apache Big Data ecosystem, allowing existing Hadoop/Spark and Beam workloads to read or write data directly from BigQuery. BigQuery allows you to get the most out of structured data by making it easy to analyze in SQL and easy to integrate with your existing Big Data jobs, so you don't have to throw away the work you've already done.

Data Governance

BigQuery provides fine-grained access controls on data and role-based control on API through integration with [Cloud IAM](#). With BigQuery and Cloud IAM, you can be sure your data is safe from unauthorized access.

Programmatic Interaction

BigQuery provides a REST API for easy programmatic access and application integration. To enable programmers of all types, BigQuery offers client libraries in Java, Python, Node.js, C#, Go, Ruby, and PHP. Business users can use Google Apps Script to access BigQuery from Google Sheets.

Rich Monitoring and Logging with Stackdriver

BigQuery provides rich monitoring, logging, and alerting through [Stackdriver Audit Logs](#). BigQuery resources can be monitored at a glance, and BigQuery can serve as a repository for logs from any application or service using Stackdriver Logging.

Cost Controls

BigQuery provides cost control mechanisms that enable you to cap your daily costs. See more information on [cost controls](#).

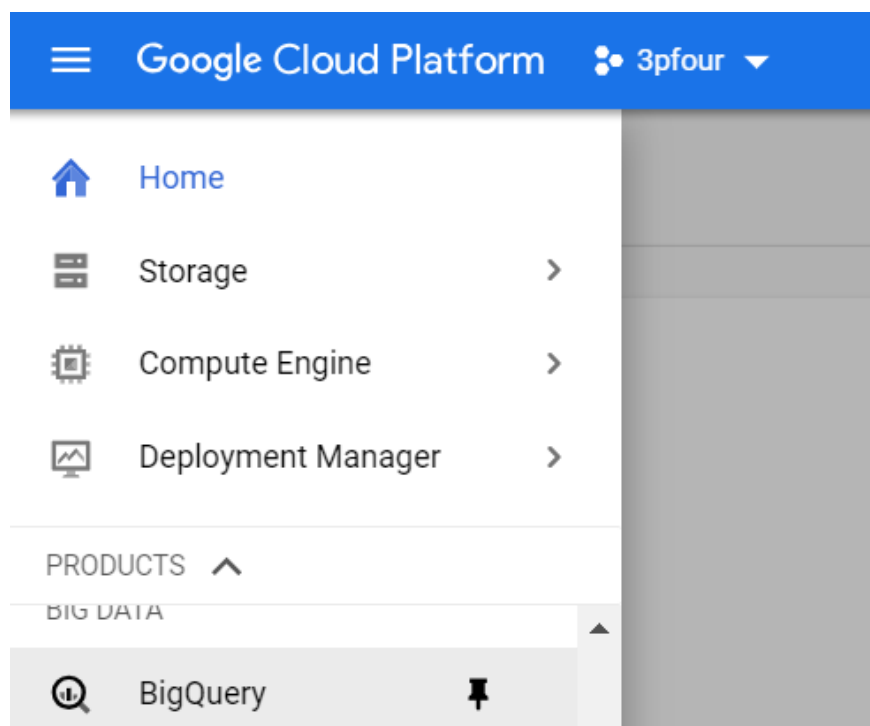
Reference: <https://cloud.google.com/bigquery/>

BigQuery Example

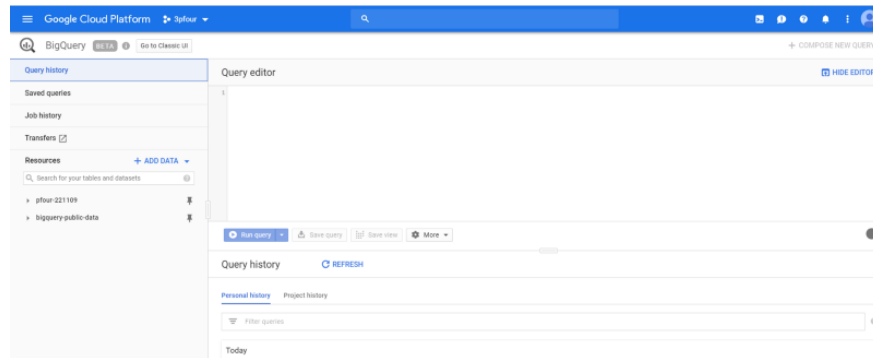
Run a Query on public dataset

First of all you must have your GCP account and billing setup ready.

Now goto BigQuery in Big Data.



You will get either a old UI or New UI based on when you open as new UI is on beta.



BigQuery

Basically speaking, this UI and CLI tool is the engine to process all your databases for Analytical purpose.

Most basic task is to load data into the storage or in Bigquery itself and run query on it.

So, First download public data from the link
<http://www.ssa.gov/OACT/babynames/names.zip>

Unzip this file and check the datas. Select some year and copy the txt file to desktop

Again goto BigQuery Web UI and navigation to **Resources** section, click your project name.

1. On the right side, in the details panel, click **Create dataset**.
2. On the **Create dataset** page:
 - For **Dataset ID**, enter `babynames` .
 - For **Data location**, choose **United States (US)**. Currently, the public datasets are stored in the `us` multi-region location. For simplicity, you should place your dataset in the same location.

Create dataset

Dataset ID

Data location (Optional) ?

Default table expiration ?

☒ Never

☐ Number of days:

Leave all of the other default settings in place and click **Create dataset**.

Load the data into a new table

Next, load the data into a new table.

1. In the navigation panel, in the **Resources** section, click the **babynames** dataset that you just created.
2. On the right side, in the details panel, click **Create table**.
3. Use the default values for all settings unless otherwise indicated.
4. On the **Create table** page:
 - For **Source Data**, click **Empty table** and choose **Upload**.
 - For **Select file**, click **Browse**, navigate to the `yob2014.txt` file and click **Open**.
 - For **File format**, click **Avro** and choose **CSV**.
 - For **Destination table**, enter `names_2014`.
 - In the **Schema** section, click the **Edit as text** toggle and paste the following schema definition in the box.
 - `name:string,gender:string,count:integer`

Create table

Source data

Create table from: Upload Select file: yob2014.txt Browse File format: CSV

Destination table

Destination project: My Project Destination dataset: babynames Table type: Native Table

Destination table

names_2014

Schema

Auto detect

☐ Schema and input parameters

☒ Edit as text

1 name:string,gender:string,count:integer

1. Click **Create Table**.
2. Wait for BigQuery to create the table and load the data. While BigQuery loads the data, a **(1 running)** string displays beside the job history in the navigation panel. The string disappears after the data is loaded.

Preview the table

After the **(1 running)** string disappears, you can access the table. To preview the first few rows of the data:

1. Select **babynames > names_2014** in the navigation panel.
2. In the details panel, click the **Preview** tab.

names_2014 [QUERY TABLE](#) [COPY TABLE](#) [DELETE TABLE](#) [EXPORT](#)

Schema Details **Preview**

Row	name	gender	count
1	Andi	F	256
2	Cristina	F	256
3	Bayleigh	F	256
4	Princess	F	256
5	Noor	F	256

Query the table

Now that you’ve loaded data into a table, you can run queries against it. The process is identical to the previous example, except that this time, you’re querying your table instead of a public table.


1.


If necessary, click the **Compose new query** button. Unless you hid the query window previously, it should still be visible.
2.

Copy and paste the following query into the query text area. This query retrieves the top 5 baby names for US males in 2014.
- ```
SELECT
 name, count
FROM
 `babynames.names_2014`
WHERE
 gender = 'M'
ORDER BY count DESC LIMIT 5
```

Click **Run query**. The results are displayed below the query window.

Query results

 **SAVE AS** ▼

 **EXPLORE IN DATA STUDIO**

Query complete (0.945 sec elapsed, 621.82 KB processed)

Job information

**Results**

JSON

Execution details

| Row | name    | count |
|-----|---------|-------|
| 1   | Noah    | 19263 |
| 2   | Liam    | 18440 |
| 3   | Mason   | 17177 |
| 4   | Jacob   | 16842 |
| 5   | William | 16798 |

Query result

=====

==

# Cloud Spanner



Cloud Spanner

***Cloud Spanner is a fully managed, mission-critical relational database system.*** It has special properties like

1. High Availability
2. Transactional consistency
3. global scale

With traditional relational semantics like schemas, ACID transaction, SQL with automatic and synchronous replication cloud spanner is the only product of its kind in the market.

Cloud spanner is first horizontally scalable. strongly consistent, relational database service.

Cloud Spanner is the only enterprise-grade, globally-distributed, and strongly consistent database service built for the cloud specifically to combine the benefits of a relational database structure with non-relational horizontal scale. This combination delivers high-performance transactions and strong consistency across rows, regions, and continents with an industry-leading 99.999% availability SLA, no planned downtime, and enterprise-grade security. Cloud Spanner revolutionizes database administration and management and makes application development more efficient.

In today's always-on, globally-distributed world, IT and developer efficiency, measured in the app downtime and time to market, is one of an organization's most precious resources. The challenge of efficiently managing app database backends while at the same time giving developers the tools they need to build efficiently was previously a challenge.

Cloud Spanner: The best of the relational and non-relational worlds

|              | CLOUD SPANNER | TRADITIONAL RELATIONAL | TRADITIONAL NON-RELATIONAL |
|--------------|---------------|------------------------|----------------------------|
| Schema       | ✓ Yes         | ✓ Yes                  | ✗ No                       |
| SQL          | ✓ Yes         | ✓ Yes                  | ✗ No                       |
| Consistency  | ✓ Strong      | ✓ Strong               | ✗ Eventual                 |
| Availability | ✓ High        | ✗ Failover             | ✓ High                     |
| Scalability  | ✓ Horizontal  | ✗ Vertical             | ✓ Horizontal               |
| Replication  | ✓ Automatic   | ⚙️ Configurable        | ⚙️ Configurable            |

References: <https://cloud.google.com/spanner/>

## Use Cases

Customers across industries can use Cloud Spanner to deliver value to their customers:

| USE CASE                                  | BEFORE CLOUD SPANNER                                                                                                             | WITH CLOUD SPANNER                                                                             |
|-------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------|
| Financial trading                         | Inconsistencies lead to potential monetary loss during reconciliation. Global synchronous replication of trades is not feasible. | Cost savings and a consistent, unified, global view.                                           |
| Insurance                                 | Inconsistencies lead to incomplete views of customers.                                                                           | Up-to-date customer views provide more accurate, real-time data.                               |
| Global call centers                       | Eventual and out-of-touch.                                                                                                       | Real-time and up-to-date.                                                                      |
| Supply-chain management and manufacturing | Global supply chain presents an inconsistent global view and/or data must be shipped in batches.                                 | Global, real-time, consistent view enables real-time decision making.                          |
| Telecom and billing                       | Processing capacity limited to finite scale-up compute resources.                                                                | Scale-out allows improved processing speed.                                                    |
| Logistics and Transportation              | Regional reach with many systems glued together.                                                                                 | Global reach with lower latency and a consistent view.                                         |
| Gaming                                    | Each server or cluster is its own universe.                                                                                      | Consistent, global view delivers a unified experience.                                         |
| E-Commerce (High Availability)            | Limited availability SLA or no SLA guarantees. In practice, potential missed sales.                                              | Guaranteed max of 5 minutes of downtime (including planned downtime) on paper and in practice. |

References: <https://cloud.google.com/spanner/>

## CLOUD SPANNER FEATURES

The first horizontally scalable, globally consistent, relational database service

### Global Scale

Horizontally scalable across rows, regions, and continents, from 1 to hundreds or thousands of nodes.

### Fully Managed

Ease of deployment at every scale and every stage. Synchronous replication and maintenance are automatic and built-in.

### Relational Semantics

Everything you would expect from a relational database—schemas, ACID transactions, and [SQL queries](#) (ANSI 2011).

### Multi-Language Support

[Client libraries](#) in C#, Go, Java, Node.js, PHP, Python, and Ruby. [JDBC driver](#) for connectivity with popular third-party tools.

### Transactional Consistency

Purpose-built for [external](#), strong, global transactional consistency.

### Enterprise Grade Security

Data-layer encryption, [IAM integration](#) for access and controls, and [audit logging](#).

### Highly Available

Whenever, wherever, your data is [highly available](#).

References: <https://cloud.google.com/spanner/>

## Example : Cloud Spanner

Cloud spanner is only used when you need better scalability and higher availability.

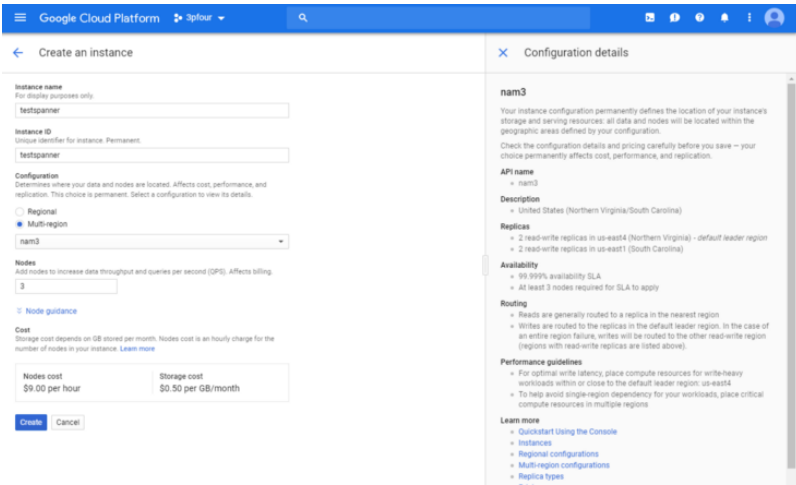
This embedded content is from a site that does not comply with the Do Not Track (DNT) setting now enabled on your browser.

Please note, if you click through and view it anyway, you may be tracked by the website hosting the embed.

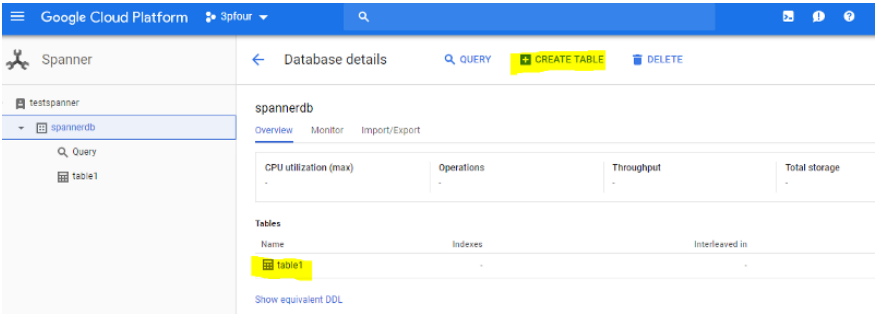
Learn More about Medium's DNT policy

# Creating an Instance

In Cloud Spanner compute nodes and storage are separated to enable scaling each independently. In addition, your nodes and data are replicated across multiple zones or even regions for high availability.

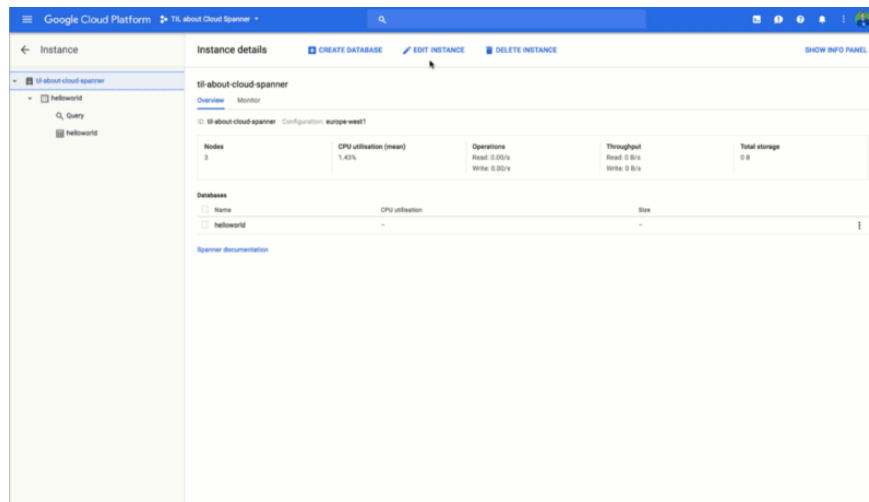


Further , I created table and column in database



## Scaling an Instance

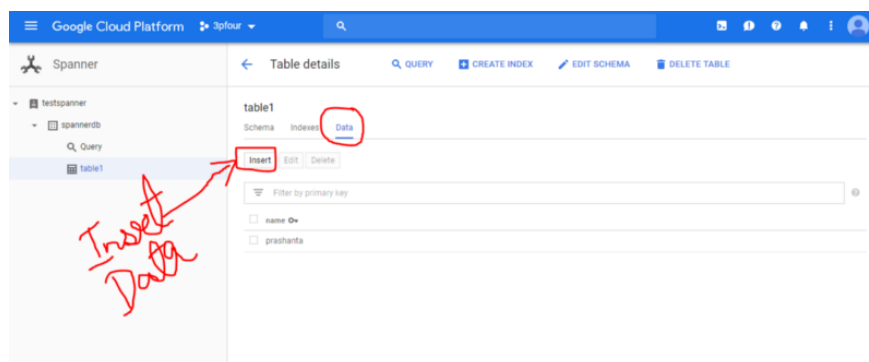
For each node you request for your instance you get compute resources in all zones and regions depending on your selected configuration. There is no need for difficult planning and forecasting on how many resources to provision upfront. You can just change the number of instance nodes any time you need more or fewer.



It is easy to scale your Cloud Spanner Instance!

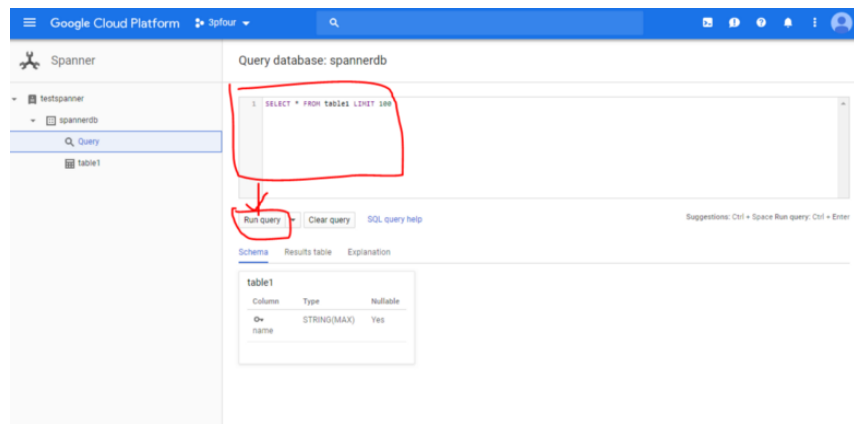
## Reading and Writing Data

Reading and writing can be done from client machines and through API.



Insert Data

Query Data



Query Data

=====

=====

## CLOUD PUB/SUB

**Ingest event streams from anywhere, at any scale, for simple, reliable, real-time stream analytics**

## Deliver event data wherever you need it

Cloud Pub/Sub is a simple, reliable, scalable foundation for stream analytics and event-driven computing systems. As part of Google Cloud's stream analytics solution, the service ingests event streams and delivers them to Cloud Dataflow for processing and BigQuery for analysis as a data warehousing solution. Relying on the Cloud Pub/Sub service for delivery of event data frees you to focus on transforming your business and data systems with applications such as:

- Real-time personalization in gaming
- Fast reporting, targeting and optimization in advertising and media
- Processing device data for healthcare, manufacturing, oil and gas, and logistics



- Syndicating market-related data streams for financial services

## Build multi-cloud and hybrid applications on open architecture

Syndicate data across projects and applications running on other clouds, or between cloud and on-premises apps. Cloud Pub/Sub easily fits in your existing environment via efficient client libraries for multiple languages, open REST/HTTP and gRPC service APIs, and an open source Apache Kafka connector.

## Scale responsively and automatically

Scale to hundreds of millions of messages per second and pay only for the resources you use. There are no partitions or local instances to manage, reducing operational overhead. Data is automatically and intelligently distributed across data centers over our unique, high-speed private network.

## Bring reliability and security tools to real-time apps

Use Cloud Pub/Sub to simplify scalable, distributed systems. All published data is synchronously replicated across availability zones to ensure that messages are available to consumers for processing as soon as they are ready. Fine-grained access controls allow for sophisticated cross-team and organizational data sharing. And end-to-end encryption adds security to your pipelines.

### CLOUD PUB/SUB FEATURES

#### At-least-once delivery

Synchronous, cross-zone message replication and per-message receipt tracking ensures at-least-once delivery at any scale.

#### Exactly-once processing

Cloud Dataflow supports reliable, expressive, exactly-once processing of Cloud Pub/Sub streams.

#### No provisioning, auto-everything

Cloud Pub/Sub does not have shards or partitions. Just set your quota, publish and consume.

#### Integrated

Take advantage of integrations with multiple services, such as Cloud Storage and Gmail update events and Cloud Functions for serverless event-driven computing.

#### Open

Open APIs and client libraries in seven languages support cross-cloud and hybrid deployments.

#### Global by default

Publish from anywhere in the world and consume from anywhere, with consistent latency. No replication necessary.

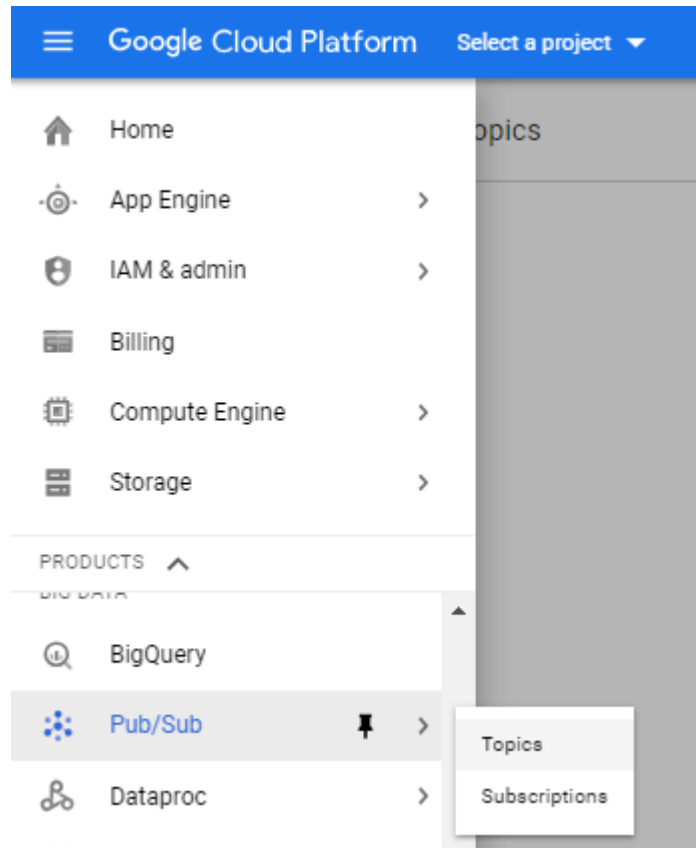
#### Compliance & security

Cloud Pub/Sub is a HIPAA-compliant service, offering fine-grained access controls and end-to-end encryption.

Ref: <https://cloud.google.com/pubsub/>

Example : Cloud Pub/Sub

Starting cloud pub/sub



## Cloud Pub/Sub Tutorial

This simple tutorial demonstrates writing, deploying, and triggering a [Background Cloud Function](#) with a [Cloud Pub/Sub trigger](#). To learn more about Cloud Pub/Sub, see the Cloud Pub/Sub [documentation](#).

**Note:** You can also use [HTTP-triggered functions](#) to listen to [Pub/Sub push subscriptions](#). This allows a single Cloud Function to subscribe to multiple topics.

## Objectives

- Write and deploy a [Background Cloud Function](#).

- Trigger the function by publishing a message to a Cloud Pub/Sub topic.

## Preparing the application

1. Clone the sample app repository to your local machine:
  - PYTHON (BETA)
  - `git clone https://github.com/GoogleCloudPlatform/python-docs-samples.git`
1. Alternatively, you can download the sample as a zip file and extract it
2. Change to the directory that contains the Cloud Functions sample code for accessing Cloud Pub/Sub:
  - PYTHON (BETA)
  - `cd python-docs-samples/functions/helloworld/`
1. Take a look at the sample code:
  - PYTHON (BETA)
1. [functions/helloworld/main.py](#)
2. [VIEW ON GITHUB FEEDBACK](#)
  - ```
def hello_pubsub(data, context):
    """Background Cloud Function to be triggered by Pub/Sub.
    Args:
        data (dict): The dictionary with data specific to this type of event.
        context (google.cloud.functions.Context): The Cloud Functions
        event
        metadata.
    """
    import base64

    if 'data' in data:
        name = base64.b64decode(data['data']).decode('utf-8')
    else:
```

```
name = 'World'
print('Hello, {}'.format(name))
```

Deploying the function

To deploy the function with a Cloud Pub/Sub trigger, run the following command in the directory where the sample code is located:

PYTHON (BETA)

```
gcloud functions deploy hello_pubsub --runtime python37 --
trigger-resource YOUR_TOPIC_NAME --trigger-event
google.pubsub.topic.publish
```

where `YOUR_TOPIC_NAME` is the name of the Cloud Pub/Sub topic to which the function will be subscribed.

Note: The function deployment creates the specified Cloud Pub/Sub topic for you. You can also specify an existing topic in your project to subscribe to.

Triggering the function

1. Publish a message to your Cloud Pub/Sub topic. In this example, the message is a name that the function will include in a greeting:
 - `gcloud pubsub topics publish YOUR_TOPIC_NAME --message YOUR_NAME`
1. Replace `YOUR_TOPIC_NAME` with the name of your Cloud Pub/Sub topic, and `YOUR_NAME` with an arbitrary string.
2. Check the logs to be sure the executions have completed:
 - `gcloud functions logs read—limit 50`
1. **Note:** Logs might take a few moments to appear. If you don't see them immediately, check again in a minute or two.

. . .

Cleaning up

To avoid incurring charges to your Google Cloud Platform account for the resources used in this tutorial:

Deleting the project

The easiest way to eliminate billing is to delete the project you created for the tutorial.

=====

Cloud Bigtable




Cloud Bigtable

Cloud Bigtable is a fully managed NoSQL database service that can scale up to petabyte for analytical and operational workloads. It has very low latency as small as 10ms. It has replication option in case of zonal failures.

The most widely used area for Bigtable is Machine Learning and AI.

It is easily integrable with current Big data tools like Hadoop and HBase.

CLOUD BIGTABLE BENEFITS

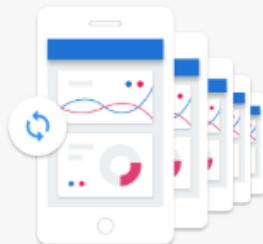


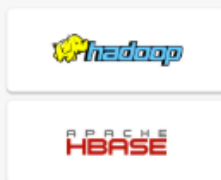
Fast and performant

Use Cloud Bigtable as the storage engine for large-scale, low-latency applications as well as throughput-intensive data processing and analytics.

Seamless scaling and replication

Provision and scale to hundreds of petabytes, and smoothly handle millions of operations per second. Changes to the deployment configuration are immediate, so there's no downtime during reconfiguration. Replication adds high availability for live serving apps, and workload isolation for serving vs. analytics.






Simple and integrated

Cloud Bigtable integrates easily with popular big data tools like [Hadoop](#), [Cloud Dataflow](#), and [Cloud Dataproc](#). Plus, Cloud Bigtable supports the open source industry standard [HBase API](#), which makes it easy for your development teams to get started.

Fully managed

Because we manage the database and handle the configuring and tuning, you can focus on developing applications.



Reference: <https://cloud.google.com/bigtable/>

Use Cases

Cloud Bigtables are used in Ad tech, Fintech, and IoT

Quickstart Using cbt

This page explains how to use the `cbt` command to connect to a Cloud Bigtable instance, perform basic administrative tasks, and read and

write data in a table.

If you are familiar with HBase, you might want to follow the [quickstart using the HBase shell](#) instead.

Create a Cloud Bigtable instance

1. Open the Create Instance page in the Google Cloud Platform Console.
2. [OPEN THE CREATE INSTANCE PAGE](#)

A Cloud Bigtable instance is a container for your clusters. [Learn more](#)

Instance name

For display purposes only

Instance ID

ID is permanent

Instance type ?

☐

Production (recommended)

Minimum of 3 nodes. High availability. Cannot downgrade later.

☒

Development

Low-cost instance for development and testing. Does not provide high availability or replication. Can upgrade to Production later.

Storage type ?

Choice is permanent. Applies to all clusters. Affects cost.

☒

SSD

Lower latency and higher read QPS. Typically used for real-time serving use cases, such as ad serving and mobile app recommendations.

☐

HDD

Higher latency for random reads. Good performance on scans and typically used for batch analytics, such as machine learning or data mining.

Clusters

Edit item ^

Cluster ID
ID is permanent.

1. For **Instance name**, enter `Quickstart instance` .
2. For **Instance ID**, enter `quickstart-instance` .

3. For **Instance type**, select **Development**.
4. For **Storage type**, select **SSD**.
5. For **Cluster ID**, enter `quickstart-instance-c1`.
6. For **Region**, select **us-east1**.
7. For **Zone**, select **us-east1-c**.
8. Click **Create** to create the instance.

Connect to your instance

1. [Install the Cloud SDK](#) if you haven't already.
2. **Note:** You can also connect using Cloud Shell, which comes with the Cloud SDK preinstalled.
3. Open a terminal window, either locally or with Cloud Shell:
4. [OPEN CLOUD SHELL](#)
5. Install the `cbt` command:
 - ```
gcloud components update
gcloud components install cbt
```
1. Configure `cbt` to use your project and instance by creating a `.cbtrc` file, replacing `[PROJECT_ID]` with the project ID in which you created your Cloud Bigtable instance:
  - ```
echo project = [PROJECT_ID] > ~/.cbtrc
echo instance = quickstart-instance >> ~/.cbtrc
```

Now you can use the `cbt` command with your instance!

Read and write data

Cloud Bigtable stores data in *tables*, which contain *rows*. Each row is identified by a *row key*.

Data in a row is organized into *column families*, or groups of columns. A *column qualifier* identifies a single column within a column family.

A *cell* is the intersection of a row and a column. Each cell can contain multiple *versions* of a value.

1. Create a table named `my-table` .

- `cbt createtable my-table`

1. List your tables:

- `cbt ls`

1. The command displays output similar to the following:

- `my-table`

1. Add one column family named `cf1` :

- `cbt createfamily my-table cf1`

1. List your column families:

- `cbt ls my-table`

1. The command displays output similar to the following:

- `cf1`

1. Put the value `test-value` in the row `r1` , using the column family `cf1` and the column qualifier `c1`

- `cbt set my-table r1 cf1:c1=test-value`

1. Use the `cbt read` command to read the data you added to the table:

- `cbt read my-table`

1. The shell displays output similar to the following:

- ```

r1
cf1:c1 @ 2016/10/31-15:05:38.840000
"test-value"
```

1. Delete the table `my-table`

- `cbt deletetable my-table`

. . .

## Clean up

To avoid incurring charges to your Google Cloud Platform account for the resources used in this quickstart:

1. Open the list of Cloud Bigtable instances in the GCP Console.
2. [OPEN THE INSTANCE LIST](#)
3. Click **Quickstart instance**.
4. Click **Delete instance**.

### Delete instance and cluster?

Deleting this instance will permanently remove the instance, its cluster, and all the cluster's data. You cannot undo this later.

Confirm deletion by typing the instance ID below: quickstart-instance

[CANCEL](#) [DELETE](#)

1. Type `quickstart-instance` , then click **Delete** to delete the instance.

2. In your terminal, delete the `.cbtrc` file:

- `rm ~/.cbtrc`

=====

=====

In this way we finished 3.4



