



<codoa
codo/>

Temas

¿Qué es firestore?

Estructura de firestore

Crear Base de datos en firebase

Agencia de
Aprendizaje
a lo largo
de la vida

¿Qué es Firestore?



Cloud Firestore

Firestore es una base de datos muy flexible y fácil de usar para el desarrollo de móviles, web y servidores. Si estás familiarizado con la base de datos en tiempo real de Firebase, Firestore tiene muchas similitudes, pero con una API diferente (posiblemente más declarativa).

Obtenga fácilmente datos en tiempo real

Al igual que la base de datos en tiempo real de Firebase, Firestore proporciona métodos útiles como `.onSnapshot()`, que hacen que sea muy fácil escuchar las actualizaciones de tus datos en tiempo real. Esto hace que Firestore sea una opción ideal para los proyectos que dan prioridad a la visualización y el uso de los datos más recientes (aplicaciones de chat, por ejemplo).

Flexibilidad como base de datos NoSQL

Firestore es una opción muy flexible para backend, porque es una base de datos NoSQL. NoSQL significa que los datos no se almacenan en tablas y columnas como lo haría una base de datos SQL estándar. Está estructurada como un almacén de valores clave, como si fuera un gran objeto de JavaScript.

En otras palabras, no hay ningún esquema, ni necesidad de describir qué datos almacenará nuestra base de datos. Mientras proporcionemos claves y valores válidos, Firestore los almacenará.

Escalable sin esfuerzo

Una de las grandes ventajas de elegir Firestore, para su base de datos es la potente infraestructura sobre la que se asienta y que le permite escalar su aplicación con gran facilidad. Tanto vertical como horizontalmente. No importa si tienes cientos o millones de usuarios. Los servidores de Google serán capaces de soportar cualquier carga que les impongas.

En resumen, Firestore es una gran opción para aplicaciones tanto pequeñas como

grandes. Para aplicaciones pequeñas es potente, porque podemos hacer mucho sin mucha configuración y crear proyectos rápidamente con ellos. Firestore es muy adecuado, para proyectos grandes debido a su escalabilidad.

Configuración de Firestore en un proyecto JavaScript

Vamos a utilizar el SDK de Firestore para JavaScript. A lo largo de esta hoja de trucos, cubriremos cómo utilizar Firestore dentro del contexto de un proyecto de JavaScript. A pesar de esto, los conceptos que cubriremos aquí son fácilmente transferibles a cualquiera de las librerías cliente de Firestore disponibles.

Para empezar con Firestore, nos dirigiremos a la consola de Firebase. Puedes visitarla yendo a firebase.google.com. Tendrás que tener una cuenta de Google para iniciar sesión.

- Base de datos de documentos sin servidor que se escala fácilmente para ajustarse a cualquier demanda, sin necesidad de mantenimiento.
- Agiliza el desarrollo de aplicaciones móviles, web y del Internet de las cosas con conectividad directa a la base de datos.
- La sincronización en directo integrada y el modo sin conexión facilitan el desarrollo de aplicaciones en tiempo real.
- La seguridad totalmente personalizable y las reglas de validación de datos garantizan que la información esté siempre protegida.

Se integra a la perfección con Firebase y con servicios de Google Cloud como Cloud Functions y BigQuery.

Modelo de datos de Cloud Firestore

Cloud Firestore es una base de datos NoSQL orientada a los documentos. A diferencia de una base de datos SQL, no hay tablas ni filas; En su lugar, almacenas los datos en documentos, que se organizan en colecciones.

Cada documento contiene un conjunto de pares clave-valor. Cloud Firestore está optimizado para almacenar grandes colecciones de documentos pequeños.

Todos los documentos se deben almacenar en colecciones, y pueden contener subcolecciones y objetos anidados. Además, ambos pueden incluir campos primitivos, como strings, o tipos de objetos complejos, como listas.

Las colecciones y los documentos se crean de manera implícita en Cloud Firestore; solo debes asignar datos a un documento dentro de una colección. Si la colección o el documento no existen, Cloud Firestore los crea.

Documentos

En Cloud Firestore, la unidad de almacenamiento es el documento. Un documento es un registro que usa pocos recursos y contiene campos con valores asignados. Cada documento se identifica con un nombre.

Colección

Documento 1 - Id

- Atributo 1 - String
- Atributo 2 - Number
- Atributo 3 - String

Documento 2 - Id

- Atributo 1 - String
- Atributo 2 - Number
- Atributo 4 - Number

Un documento que representa a un usuario a lovelace puede tener el siguiente aspecto:

```
class alovelace
```

```
  first : "Ada"
```

```
  last  : "Lovelace"
```

```
  born  : 1815
```

Nota: Cloud Firestore es compatible con diversos tipos de datos para los valores, como booleanos, números, strings, puntos geográficos, BLOB binarios y marcas de tiempo. Además, puedes usar arreglos u objetos anidados, llamados mapas, para estructurar datos dentro de un documento.

Los objetos complejos anidados en un documento se llaman mapas. Por ejemplo, podrías estructurar el nombre del usuario del ejemplo anterior con un mapa como este:

```
class alovelace
```

```
  name :
```

```
    first : "Ada"
```

```
    last  : "Lovelace"
```

```
  born  : 1815
```

Tal vez te parezca que los documentos son muy similares a JSON; de hecho, básicamente son JSON. Existen algunas diferencias (por ejemplo, los documentos admiten tipos de datos adicionales y su tamaño se limita a 1 MB), pero en general, puedes tratar los documentos como registros JSON livianos.

Colecciones

Los documentos viven en colecciones, que simplemente son contenedores de documentos. Por ejemplo, podrías tener una colección llamada users con los distintos usuarios de tu app, en la que haya un documento que represente a cada uno:

```
collections_bookmark users
```

```
class alovelace
```

```
first : "Ada"
```

```
last : "Lovelace"
```

```
born : 1815
```

```
class aturing
```

```
first : "Alan" last :
```

```
"Turing" born :
```

```
1912
```

Cloud Firestore no usa esquemas, por lo que tienes libertad total sobre los campos que

pones en cada documento y los tipos de datos que almacenas en esos campos. Los documentos dentro de una misma colección pueden contener campos diferentes o almacenar distintos tipos de datos en esos campos. Sin embargo, se recomienda usar los mismos campos y tipos de datos en varios documentos, de manera que puedas consultarlos con mayor facilidad.

Una colección contiene solo documentos; no puede contener campos sin procesar con valores de manera directa ni tampoco otras colecciones (Consulta los datos jerárquicos para ver una explicación sobre cómo estructurar datos más complejos en Cloud Firestore).

Los nombres de documentos dentro de una colección son únicos. Puedes proporcionarte propias claves, como los ID de usuario, o puedes dejar que Cloud Firestore cree ID aleatorios de forma automática.

No es necesario "crear" ni "borrar" las colecciones; Cuando se crea el primer documento de una colección, esta pasa a existir. Si borras todos los documentos de una colección, esta deja de existir.

Referencias

Cada documento de Cloud Firestore se identifica de forma única por su ubicación dentro de la base de datos. El ejemplo anterior muestra un documento `alovelace` en la colección `users`. Para hacer referencia a esta ubicación en tu código, puedes crear una referencia a ella.

Una referencia es un objeto liviano que simplemente apunta a una ubicación en la base de datos. Puedes crear una referencia sin importar si existen datos ahí, y crearlo no ejecuta ninguna operación de red.

Nota: Las referencias a colecciones y las referencias a documentos son dos tipos distintos de referencias que permiten ejecutar diferentes operaciones. Por ejemplo, podrías usar una referencia a una colección para consultar los documentos de la colección y podrías usar una referencia a un documento para leer o escribir en un documento individual.

Datos jerárquicos

Para comprender cómo funcionan las estructuras de datos jerárquicas en Cloud Firestore, considera el siguiente ejemplo de una app de chat con mensajes y salas de chat.

Puedes crear una colección llamada rooms para almacenar diferentes salas de chat:

```
collections_bookmark rooms
```

```
class roomA
```

```
name : "my chat room"
```

```
class roomB
```

Ahora que tienes salas de chat, decide cómo almacenarás los mensajes. Es posible que no quieras almacenarlos en el documento de la sala de chat. Los documentos en Cloud Firestore deben usar pocos recursos, y una sala de chat podría contener muchos mensajes. Sin embargo, puedes crear colecciones adicionales en el documento de tu sala de chat, como subcolecciones.

Subcolecciones

La mejor manera de almacenar mensajes en este caso es usar subcolecciones. Una subcolección es una colección asociada con un documento específico.

Nota: Puedes realizar consultas en varias subcolecciones que tengan el mismo ID de

colección con las consultas a grupos de colecciones.

Puedes crear una subcolección llamada messages para cada documento de la sala que integra la colección rooms:

```
collections_bookmark rooms
```

```
class roomA
```

```
name : "my chat room"
```

```
collections_bookmark messages
```

```
class message1
```

```
from : "alex"
```

```
msg : "Hello World!"
```

```
class message
```

```
class roomB
```

Las subcolecciones te permiten estructurar datos de forma jerárquica, lo que facilita el acceso a los datos.

Los documentos de las subcolecciones también pueden contener subcolecciones, lo que te permite anidar datos en más niveles. Puedes anidar datos hasta 100 niveles de profundidad.

Pasos para crear una Base de datos en Firebase



1 Deberemos crear un proyecto

Comencemos con el
nombre de tu proyecto?

Nombre del proyecto

cac2022

✎ cac2022-c275b

Continuar



2 Nos da la opción de activar Google Analytics





Google Analytics para tu proyecto de Firebase


Google Analytics es una solución de analítica ilimitada y gratuita que permite usar la segmentación, los informes y otras funciones en Firebase Crashlytics, Cloud Messaging, In-App Messaging, Remote Config, A/B Testing y Cloud Functions.



Google Analytics habilita las siguientes funciones:

 Pruebas A/B 

 Segmentación de usuarios y orientación a ellos en los productos de Firebase 

 Usuarios que no experimentan fallas 

 Activadores de Cloud Functions basados en eventos 

 Informes ilimitados y gratuitos 

☒ Habilitar Google Analytics para este proyecto
Recomendado

[Anterior](#)

[Continuar](#)

3 Seleccionamos la cuenta y hacemos click en crear proyecto



Configurar Google Analytics

Elige o crea una cuenta de Google Analytics ?

 alejandrozapata73

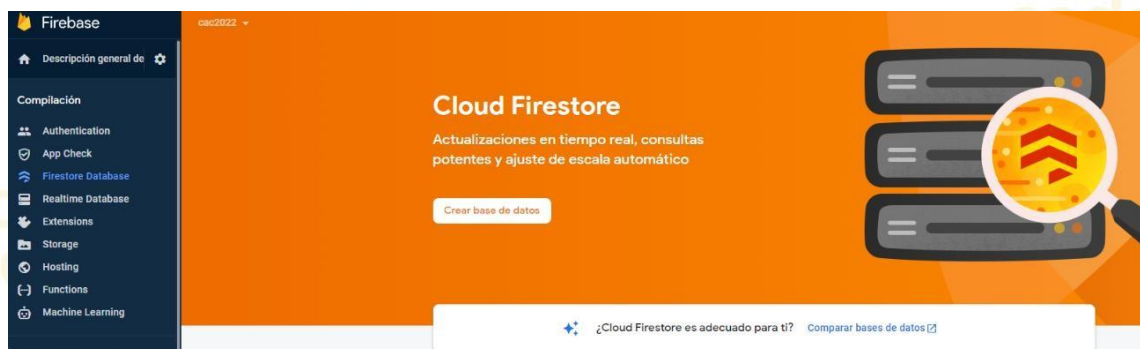
Crear automáticamente una nueva propiedad en esta cuenta 

Cuando se cree el proyecto, también se creará una nueva propiedad de Google Analytics en la cuenta de Google Analytics que elijas y se vinculará a tu proyecto de Firebase. Este vínculo permitirá el flujo de datos entre los productos. Los datos que se exportan de tu propiedad de Google Analytics a Firebase están sujetos a las Condiciones del Servicio de Firebase, mientras que los datos de Firebase que se importan a Google Analytics están sujetos a las Condiciones del Servicio de Google Analytics. [Obtén más información](#)

[Anterior](#)

[Crear proyecto](#)

4 En el panel podemos acceder a las opciones, cloud Firestore



5 Seleccionamos la opción crear base de datos

Agencia de
Aprendizaje
a lo largo
de la vida



Cloud Firestore

Actualizaciones en tiempo real, consultas potentes y ajuste de escala automático

Crear base de datos

6 Elegimos la opción modo de prueba

Crear base de datos

1 Crea reglas de seguridad de Cloud Firestore — 2 Configura la ubicación de Cloud Firestore

Después de definir la estructura de datos, debes crear reglas para protegerlos.
[Más información](#)

☐ Iniciar en modo de producción
De forma predeterminada, tus datos son privados. El acceso de lectura/escritura de los clientes solo se otorgará como se indica en tus reglas de seguridad.

☒ Comenzar en modo de prueba
Para permitir una configuración rápida, los datos se abren de forma predeterminada. Sin embargo, debes actualizar las reglas de seguridad dentro de 30 días a fin de habilitar el acceso de lectura/escritura a largo plazo para los clientes.

```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    match /{document=**} {
      allow read, write: if
        request.time < timestamp.date(2022, 7, 2);
    }
  }
}
```

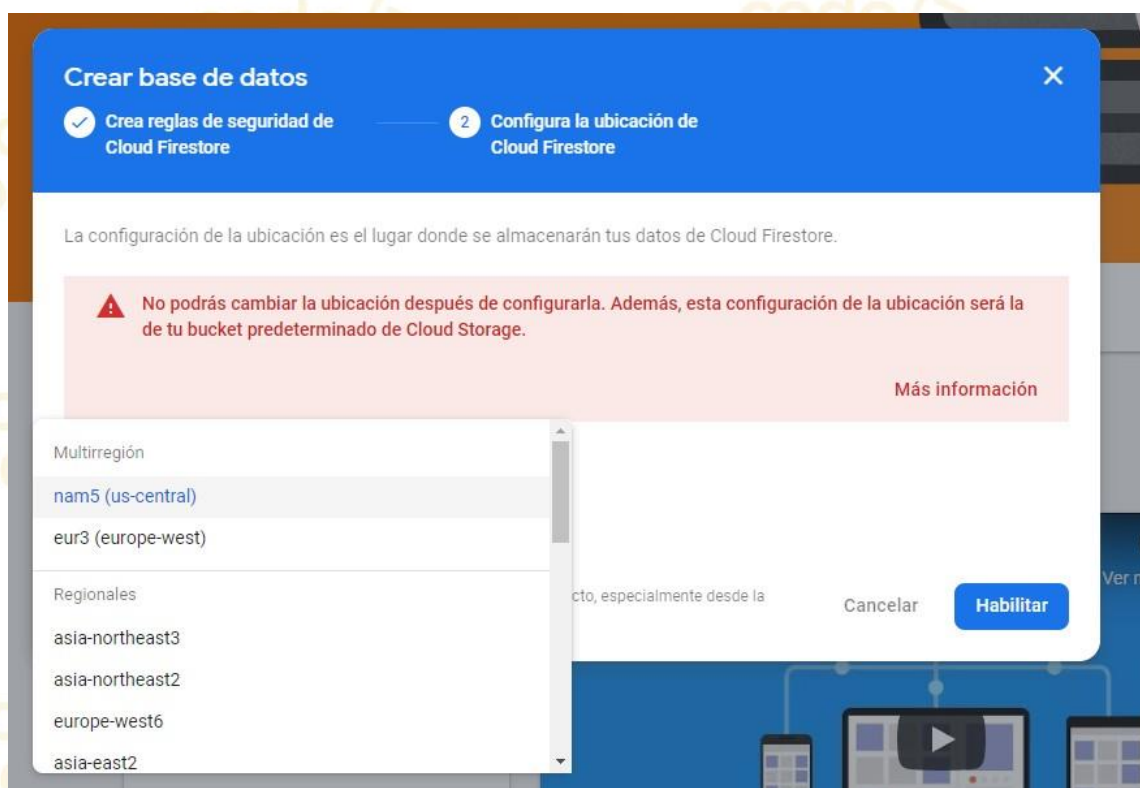
! Las reglas de seguridad predeterminadas del modo de prueba permiten que cualquier usuario con acceso a tu referencia de base de datos pueda ver, editar y borrar todos los datos durante los siguientes 30 días.

Si habilitas Cloud Firestore, no podrás usar Cloud Datastore en este proyecto, especialmente desde la aplicación de App Engine asociada

Cancelar **Siguiente**

7 Seleccionamos la zona mas próxima de los servidores

Agencia de
Aprendizaje
a lo largo
de la vida



8 Tenemos listo nuestra base de datos para poder trabajar

