
Foro 1: Base de Datos NoSQL

(Cloud Firestone y Realtime Database)

Docente

Alexander Alberto Sigüenza Campos

Integrantes


| | |
|---|---|
| Enma Sofia López Rojas | LR230079 |
| Kevin Enrique Martínez Martínez | MM230084 |
| José Gerardo Marroquín Vásquez | MV230090 |
| José Ernesto Sorto González | SG202883 |
| Roger Eduardo Vásquez Portillo | VP223250 |


Diseño y Programación de Software Multiplataforma

DPS941 G01T

San salvador 21 abril de 2023

# Contenido

# INTRODUCCIÓN

La forma en la que se almacenan y gestiona las bases de datos ha ido cambiando con el paso del tiempo, en la actualidad se cuenta con gestores de bases de datos que ya no solo se basas en datos relacionales y los cuales siguen un modelo estructurado y rígido, entre estos gestores se encuentra NoSQL, estos ofrecen flexibilidad y escalabilidad para a adaptarse a las demandas de aplicaciones modernas.

Es necesario conocer cuáles son estos gestores, así como sus características principales las cuales nos servirán de base para elegir una u otra para la ejecución de proyectos más adelante, dentro del desarrollo de la investigación se ha recopilado información acerca de Firestore y Realtime Database.

Adicional a esta información se mostrará un resumen de las características de las bases de datos SQL como NoSQL, estos nos servirán como referencia para la elección entre el tipo de gestor que se pueden utilizar en los diversos desarrollos que se puedan ejecutar.

# OBJETIVOS

**GENERAL:**

Realizar una Investigación sobre las bases de datos en entorno de desarrollo móviles

**ESPECIFICO:**

- Conocer acerca de Cloud Firestore y Realtime Database
- Identificar sus características principales.
- Hacer una tabla comparativa entre base de datos SQL y NoSQL

# MARCO TEÓRICO

## FACE I

### BASE DE DATOS NoSQL

Dentro del desarrollo del entorno móvil es necesario la aplicación de Bases de Datos, existen diferentes formas de manejo de base de dato, dentro de estas esta NoSQL (No Only SQL), este sistema de gestión de base de datos proporciona mecanismos que gestionan el almacenaje y la recuperación de datos, permitiendo así una escalabilidad y flexibilidad en el desarrollo de aplicaciones web y móviles. A continuación, conoceremos un poco más sobre este tipo de gestores de base de datos.

### ¿QUÉ ES CLOUD FIRESTORE?

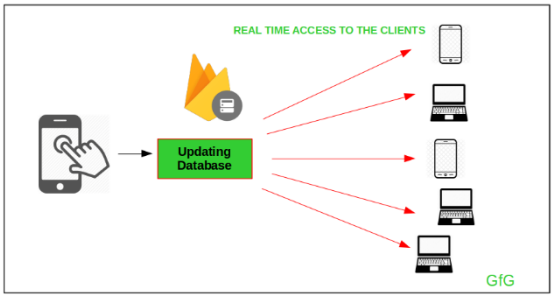

Cloud Firestore es una base de datos NoSQL alojada en la nube lo que la convierte en una base de datos flexible y escalable que permite el desarrollo móvil, web y de servidores, mediante los datos sincronizados en la aplicación cliente a través de escuchas en tiempo real, además ofrece soporte sin conexión para el desarrollo de app móviles y web. Firestore ofrece una integración perfecta con otros productos como lo son Firebase y Google Cloud, incluyendo Cloud Function. (FIREBASE, 2024)

Cloud Firestone es una base de datos NoSQL, que esta alojada en la nube donde las aplicaciones Web, Apple y Android pueden acceder mediante SDK Nativos, tanto para Node.JS, Java, Python, Unity, C++ y Go, a demás de las API REST y RPC.

La forma en que se guardan los datos dentro de Firestone es mediante documentos los cuales contienen campos asignados a valores, estos documentos se almacenan a su vez en colecciones, los

cuales forman contenedores para estos documentos, estos pueden ser organizados de diversas formas y así facilitar las consultas que se puedan realizar sobre ellos. Estos documentos permiten diferentes tipos de datos entre los que están las cadenas, números simples hasta objetos complejos y anidados, además se pueden crear sub- colecciones dentro de estos documentos, así como también permite la creación de datos jerárquicos de los cuales se pueden ir escalando según la necesidad.

## ¿QUÉ ES REALTIME DATABASE?




Firebase Realtime Database es una base de datos alojada en la nube, los datos pertenecientes a esta se guardan como JSON y se sincronizan en tiempo real con cada cliente conectado. Al crear app multiplataforma en esta plataforma de Apple, Android y SDK de JavaScript, todos los clientes comparten una instancia de Realtime Database la cual puede actualizar automáticamente con los datos más recientes. (FIREBASE, 2024)

Firebase Realtime Database permite la creación de app colaborativas y hace que el acceso sea seguro a la base de datos directamente desde el código del cliente, aun si no hay conexión los eventos en tiempo real siguen activos trabajándose desde la memoria del dispositivo y a la espera del restablecimiento de la conexión para restablecer los datos locales con las actualizaciones remotas.

Además de estas funciones también cuenta con un lenguaje de reglas flexibles basándose en expresiones llamado “Firebase Realtime Database Security Rules” las cuales establecen el orden de como y cuando deben de estructurar, leer o escribir los datos.

## DIFERENCIAS ENTRE CLOUD FIRESTORE Y REALTIME DATABASE

Para resolver esta interrogante de cuales son las diferencias entre Cloud Firestore y Realtime Database se planteará de forma comparativa las diferencias que existen entre ambas bases de datos y poder tener una vista más general de cuales son de forma clara y resumida (FIREBASE, 2024).

| Capacidades | Características | |
|---|---|---|
| | Cloud Firestone | Realtime Database |
| Modelo de datos | Almacena datos como colecciones de documentos<br>Datos simples son fácilmente almacenados en documentos similares a JSON.<br>Los datos complejos y jerárquicos son más fáciles de organizar a escalas mediante sub-colecciones.<br>Requiere menos desnormalización y apalancamiento de datos | Es una base de datos alojada en la nube.<br>Los datos son almacenados como un gran árbol JSON.<br>Los datos simples son muy fáciles de almacenar.<br>Los datos complejos y jerárquicos son más difíciles de organizar a escala. |
| Soporte en tiempo real y fuera de línea | Soporte sin conexión para Apple, Android y clientes web<br>Firestore utiliza la sincronización de datos para actualizar los datos en cualquier dispositivo. Sin embargo, esta diseñado para realizar consultas simples y únicas de manera eficiente. | Soporte sin conexión para cliente Apple y Android.<br>En lugar de la típica solicitud HTTP, Realtime Database utiliza sincronización de datos recibiendo actualizaciones en milisegundos en todos los clientes conectados |
| Fiabilidad y Rendimiento | Firestore es una solución regional y multirregional que escala automáticamente.<br>Solución de baja latencia, respuesta no superior de 30 ms.<br>Alojamiento de datos en múltiples centros de datos (distintas regiones), garantizando escalabilidad global y solida confiabilidad. | Database es una solución regional.<br>Latencia extremadamente baja, tiempos de espera no superiores a 10 ms<br>Dispone de configuraciones regionales, la base de datos está limitadas a la disponibilidad de la zona dentro de la región |
| Escalabilidad | Escalabilidad automática.<br>Escala completamente automática, los limites de escala son alrededor de 1 millon de conexiones simultaneas y 10,000 de escritura por segundo | Escalabilidad requiere Fragmentación.<br>Escala hasta alrededor de 200,000 conexiones simultanea necesita optimización manual |

| | | |
|---|---|---|
| Seguridad | Reglas sin cascada que combina autorización y validación.<br>Lee y escribe desde SDK móviles protegidos por las reglas de seguridad.<br>Lee y escribe desde SDK de servidor protegido por Identity and Access Management (IAM)<br>Las reglas no se aplican en cascadas a menos que utilice un comodín.<br>Las reglas pueden restringir las consultas: si los resultados de una consulta pueden contener datos a los que el usuario no tiene acceso, toda la consulta falla | Lenguajes de reglas en cascada que se separa autorización y validación.<br>Lee y escribe desde SDK móviles protegidos por reglas de seguridad de bases de datos en tiempo real.<br>Leer y escribir reglas en cascadas.<br>Los datos se valían por separado utilización la regla “validate” |
| Sincronización | Sincronización a nivel de documentos y colecciones | Sincronización a nivel de nodos |
| Escritura y Transacciones | Operaciones avanzadas de escritura y transacciones<br>Se escriben operaciones a través de operaciones de configuración y actualización, así como transformaciones avanzadas como operadores numéricos y matrices<br>Las transacciones pueden leer y escribir datos de forma atómicas desde cualquier parte de la base de datos | Soporta transacciones, pero con limitaciones.<br>Escribe datos mediante operaciones de configuración y actualización.<br>Las transacciones son atómicas en un árbol de datos especifico. |

Tabla 1: Tabla Comparativa de las Características de Firestone y Realtime Database.

Fuente: Elaboración Propia

En la tabla se muestran algunas de las características principales de Firestore y Realtime Database, pueden existir más, pero hemos tratado de resumirlo y mostrar ambas vistas para identificar mas detalladamente las principales

**DIFERENCIAS FUNDAMENTALES ENTRE LAS BASES DE DATOS SQL Y NOSQL**

Es bien conocido que en el ámbito del desarrollo de app se pueden hacer uso de diversos gestores de Base de Datos entre los más comunes tenemos a SQL y NoSQL, en esta ocasión se mostrara una tabla en la que se comparar las características de estas útiles herramientas. (UNIR-, 2021)

| Diferencias | Base de datos SQL | Base de Datos NoSQL |
|---|---|---|
| Almacenamiento de datos | La base de datos de SQL almacena datos estructurados; en cuestión de análisis de datos estos suelen ser más fáciles de analizar que los no estructurados. | La base de datos NoSQL almacenan los datos de forma original, es decir no estructurados; esto provoca que son más Flexibles y los cambios de arquitectura no les afectan tanto como a los estructurados. |
| Escalabilidad | Las SQL proporcionan una capacidad escalar baja en comparación de NoSQL, estas bases de datos están centralizadas. | Las Bases de datos están creadas para grandes volúmenes de información como la big data, ya que estas están bases de datos están distribuidas posibilitando su ejecución múltiple maquinas |
| Consulta | SQL es un lenguaje de consulta que se comunica con bases de datos | NoSQL es un adjetivo utilizado para describir una base de datos no relacional que no requiere el lenguaje SQL |
| Transacciones | En caso de transacciones que involucren uno o más cambios en la | El soporte para cada transacción puede ser variante y dependerá de el tipo en |

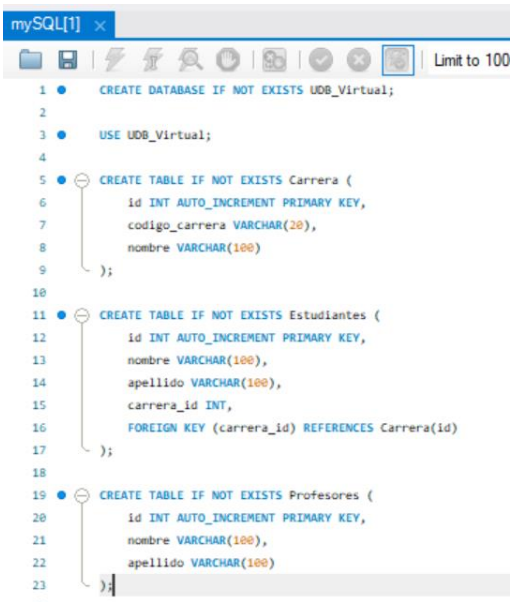
| | | |
|---|---|---|
| | base de datos, deben de realizarse como una unidad atómica, consistente, aislada y duradera (ACID por sus siglas en ingles) | especifico de datos, algunas Bases de datos NoSQL ofrecen soporte para transacciones similares a las bases de datos relacionales mientras que otras pueden tener limitaciones en este aspecto |
| Flexibilidad | Menos flexible en términos de estructura de datos | Son mas flexibles, pueden manejar dato no estructurados. |

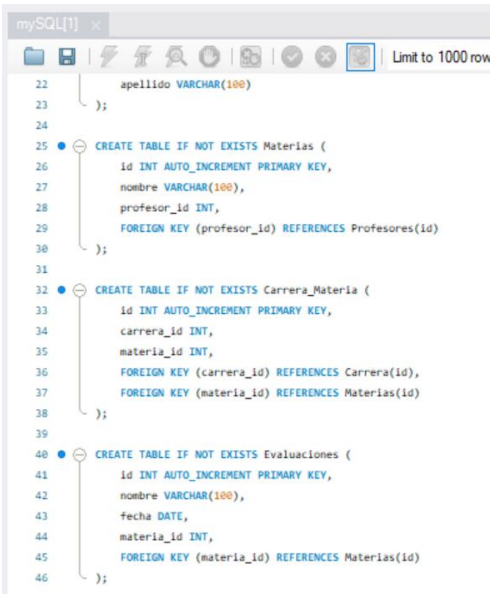
Tabla 2: Tabla Comparativa de las Características SQL y Nosql

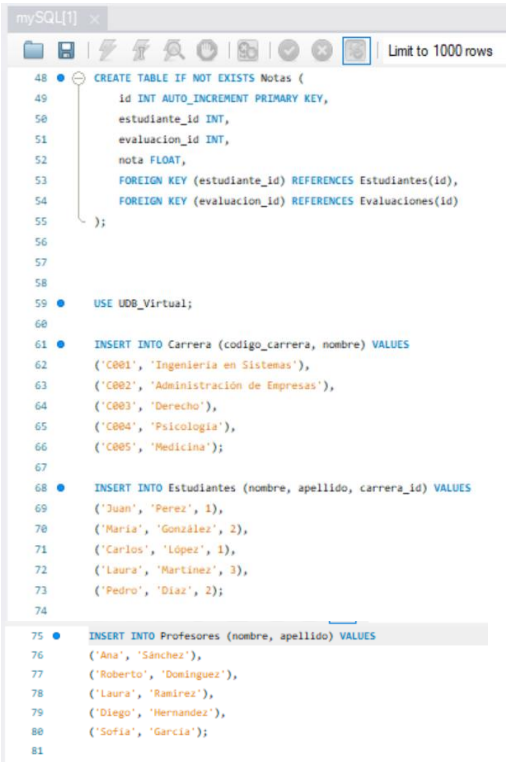
Fuente: Elaboración Propia

## FASE II

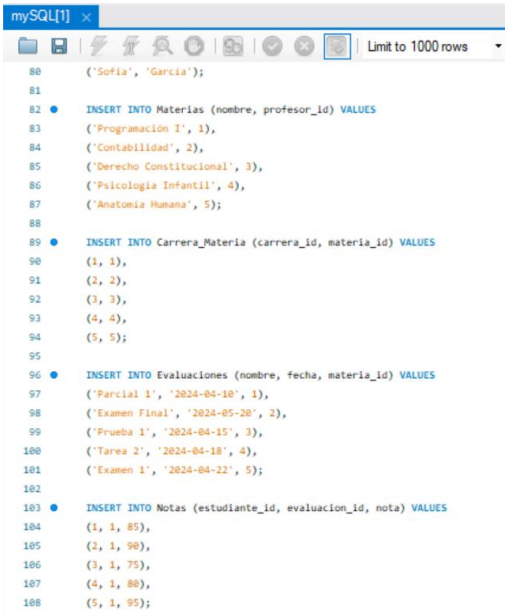
### ESTRUCTURA DE BASE DE DATOS UTILIZANDO SQL



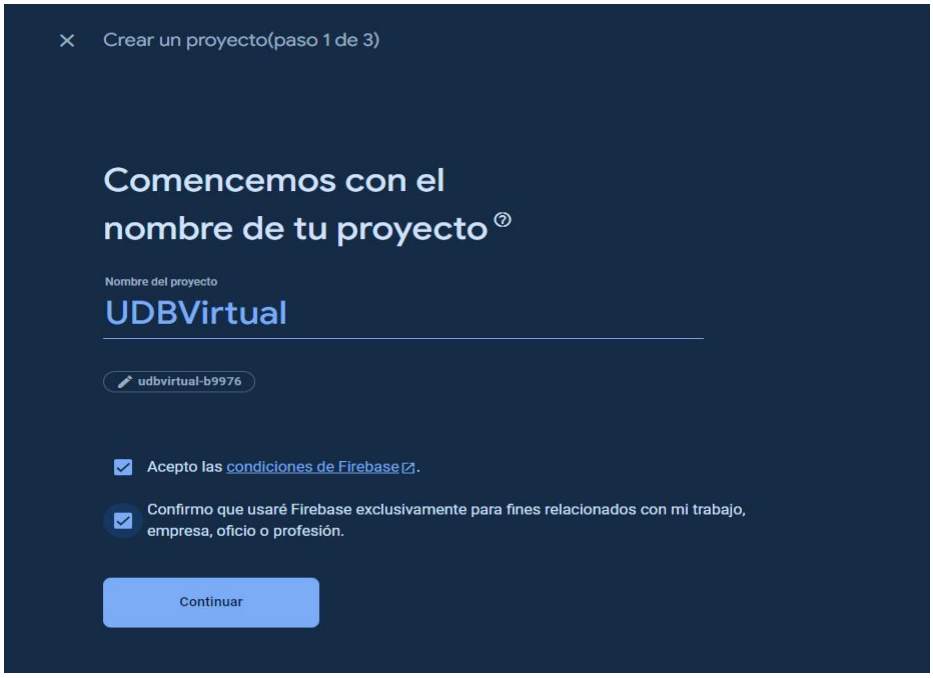


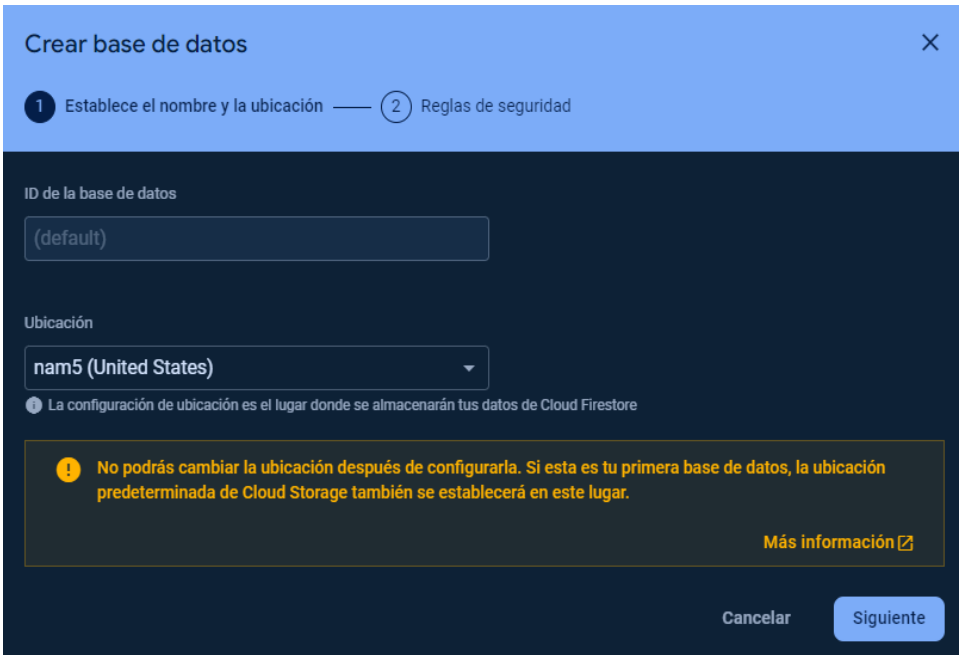
```
mySQL[1]
Limit to 1000 rows
48  CREATE TABLE IF NOT EXISTS Notas (
49      id INT AUTO_INCREMENT PRIMARY KEY,
50      estudiante_id INT,
51      evaluacion_id INT,
52      nota FLOAT,
53      FOREIGN KEY (estudiante_id) REFERENCES Estudiantes(id),
54      FOREIGN KEY (evaluacion_id) REFERENCES Evaluaciones(id)
55  );
56
57
58
59  USE UDB_Virtual;
60
61  INSERT INTO Carrera (codigo_carrera, nombre) VALUES
62  ('C001', 'Ingeniería en Sistemas'),
63  ('C002', 'Administración de Empresas'),
64  ('C003', 'Derecho'),
65  ('C004', 'Psicología'),
66  ('C005', 'Medicina');
67
68  INSERT INTO Estudiantes (nombre, apellido, carrera_id) VALUES
69  ('Juan', 'Perez', 1),
70  ('María', 'González', 2),
71  ('Carlos', 'López', 1),
72  ('Laura', 'Martínez', 3),
73  ('Pedro', 'Díaz', 2);
74
75  INSERT INTO Profesores (nombre, apellido) VALUES
76  ('Ana', 'Sánchez'),
77  ('Roberto', 'Domínguez'),
78  ('Laura', 'Ramírez'),
79  ('Diego', 'Hernández'),
80  ('Sofía', 'García');
81
```

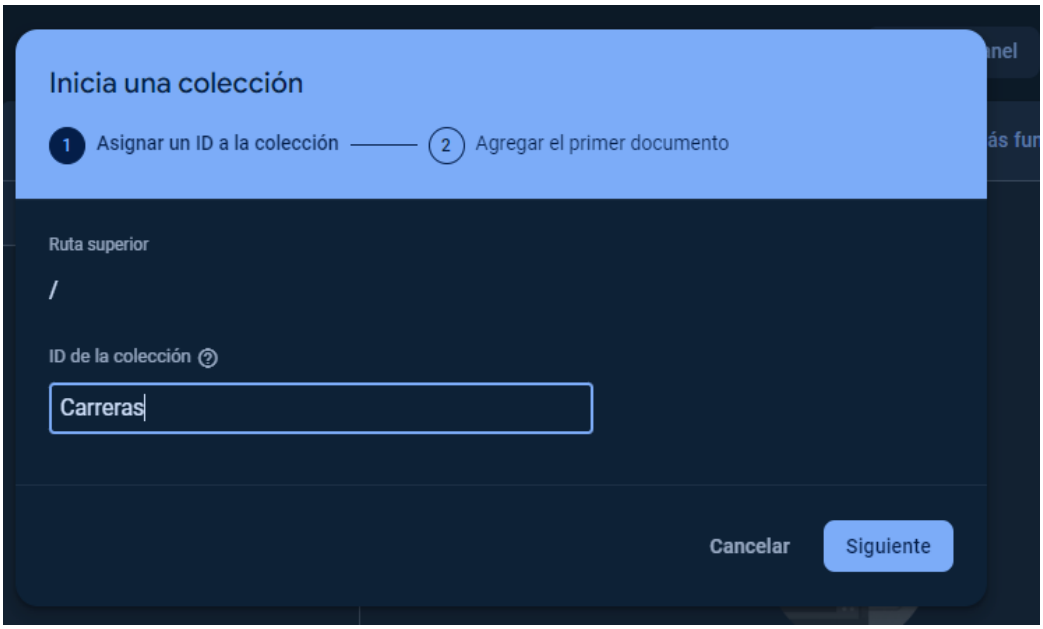


```
mySQL[1]
Limit to 1000 rows
80  ('Sofía', 'García');
81
82  INSERT INTO Materias (nombre, profesor_id) VALUES
83  ('Programación I', 1),
84  ('Contabilidad', 2),
85  ('Derecho Constitucional', 3),
86  ('Psicología Infantil', 4),
87  ('Anatomía Humana', 5);
88
89  INSERT INTO Carrera_Materia (carrera_id, materia_id) VALUES
90  (1, 1),
91  (2, 2),
92  (3, 3),
93  (4, 4),
94  (5, 5);
95
96  INSERT INTO Evaluaciones (nombre, fecha, materia_id) VALUES
97  ('Parcial 1', '2024-04-10', 1),
98  ('Examen Final', '2024-05-20', 2),
99  ('Prueba 1', '2024-04-15', 3),
100 ('Tarea 2', '2024-04-18', 4),
101 ('Examen 1', '2024-04-22', 5);
102
103 INSERT INTO Notas (estudiante_id, evaluacion_id, nota) VALUES
104 (1, 1, 85),
105 (2, 1, 90),
106 (3, 1, 75),
107 (4, 1, 80),
108 (5, 1, 95);
```

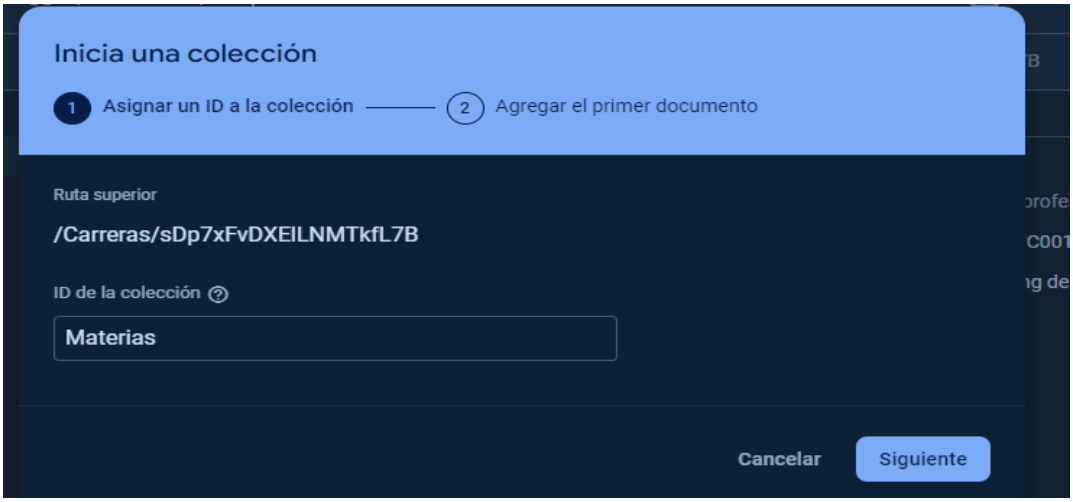
**ESTRUCTURA DE BASE DE DATOS UTILIZANDO NoSQL FIREBASE**

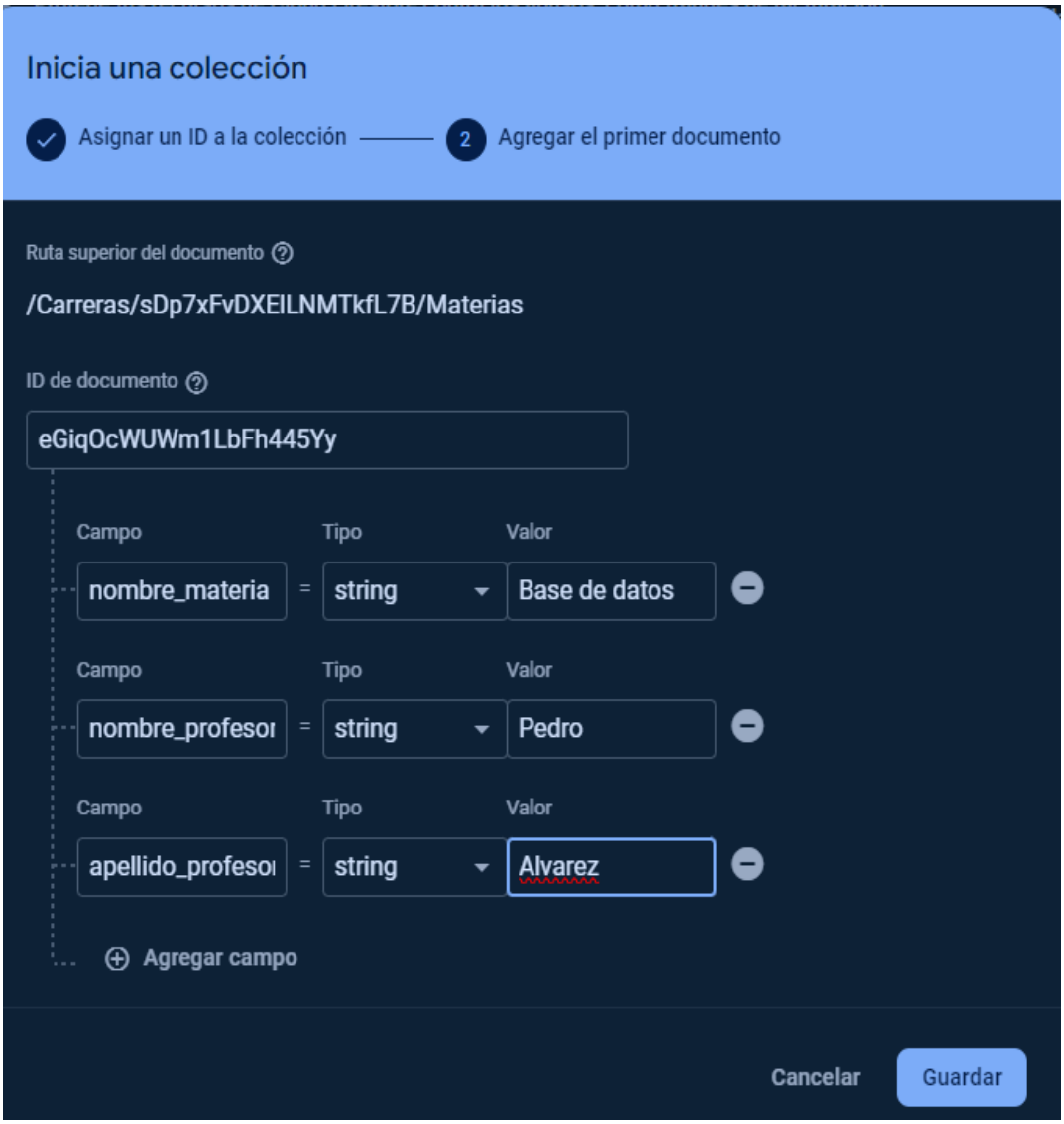


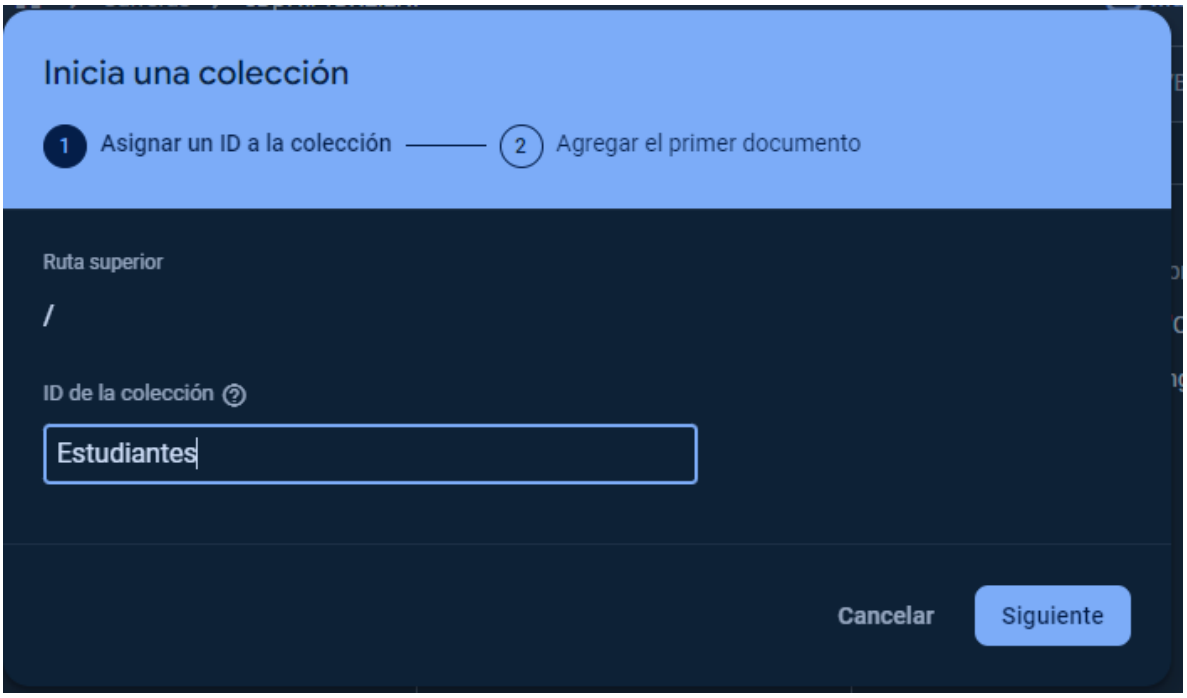

Crear base de datos
1 Establece el nombre y la ubicación
2 Reglas de seguridad
ID de la base de datos
(default)
Ubicación
nam5 (United States)
La configuración de ubicación es el lugar donde se almacenarán tus datos de Cloud Firestore
No podrás cambiar la ubicación después de configurarla. Si esta es tu primera base de datos, la ubicación predeterminada de Cloud Storage también se establecerá en este lugar.
Más información
Cancelar
Siguiente

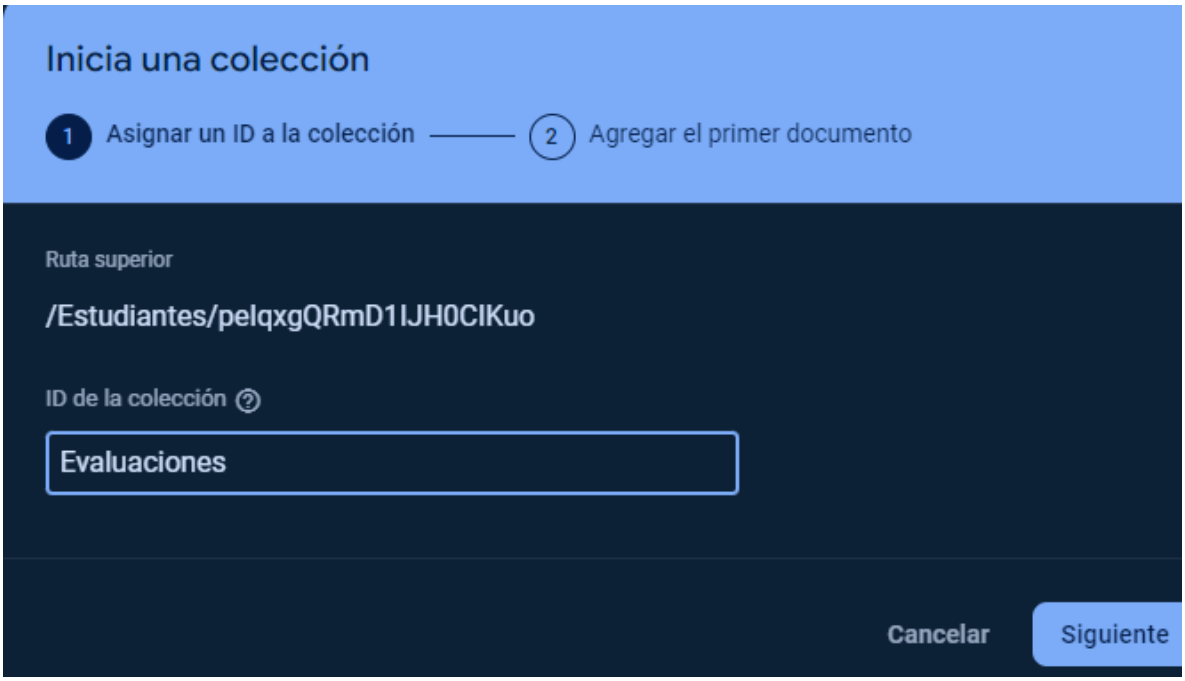

Inicia una colección
1 Asignar un ID a la colección
2 Agregar el primer documento
Ruta superior
/
ID de la colección
Carreras
Cancelar
Siguiente

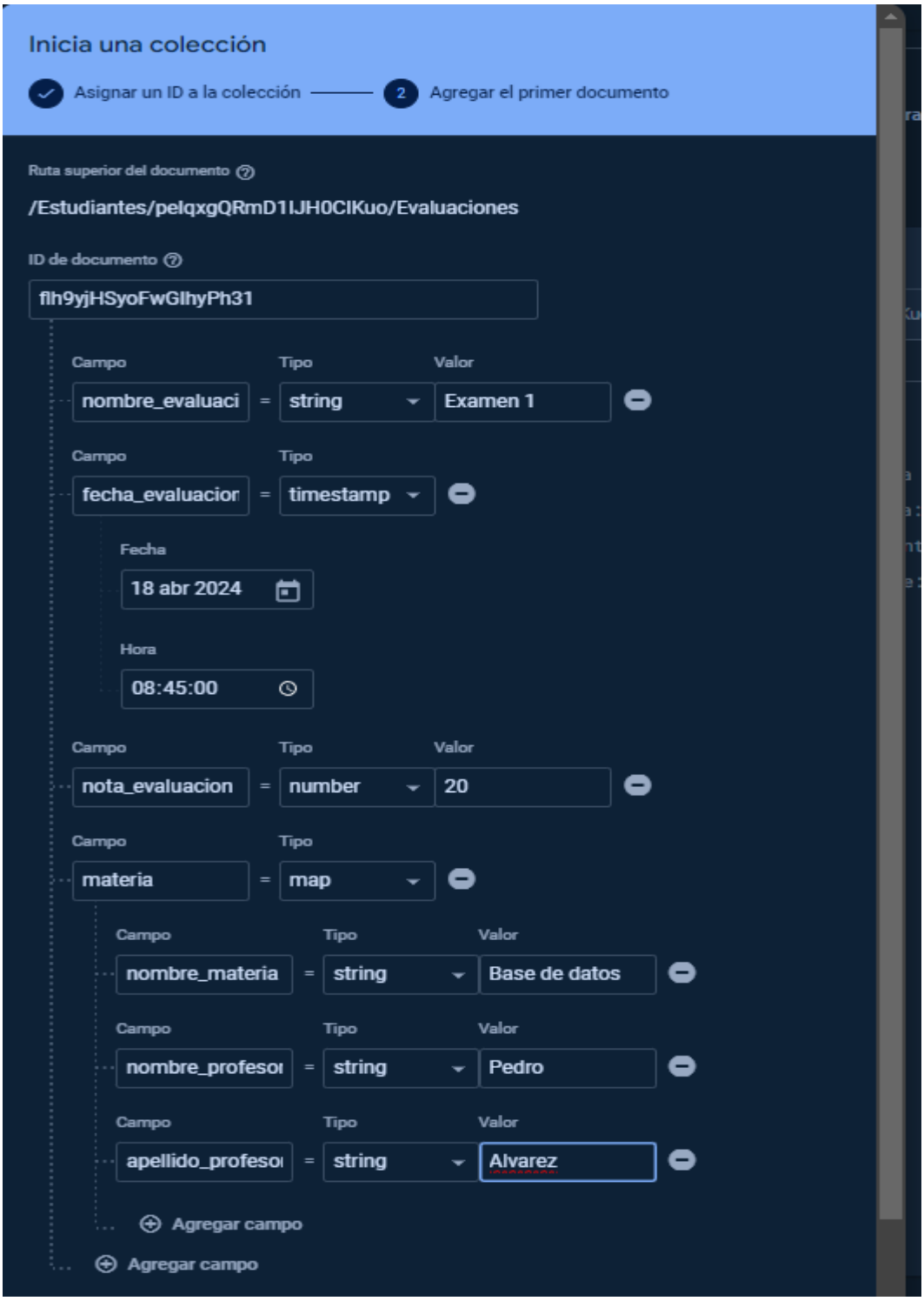

Ruta superior del documento
/Carreras
ID de documento
sDp7xFvDXElLNMTkfL7B
Campo
Tipo
Valor
codigo_carrera
=
string
C001
Campo
Tipo
Valor
nombre_carrera
=
string
Ing de Sistemas
Campo
Tipo

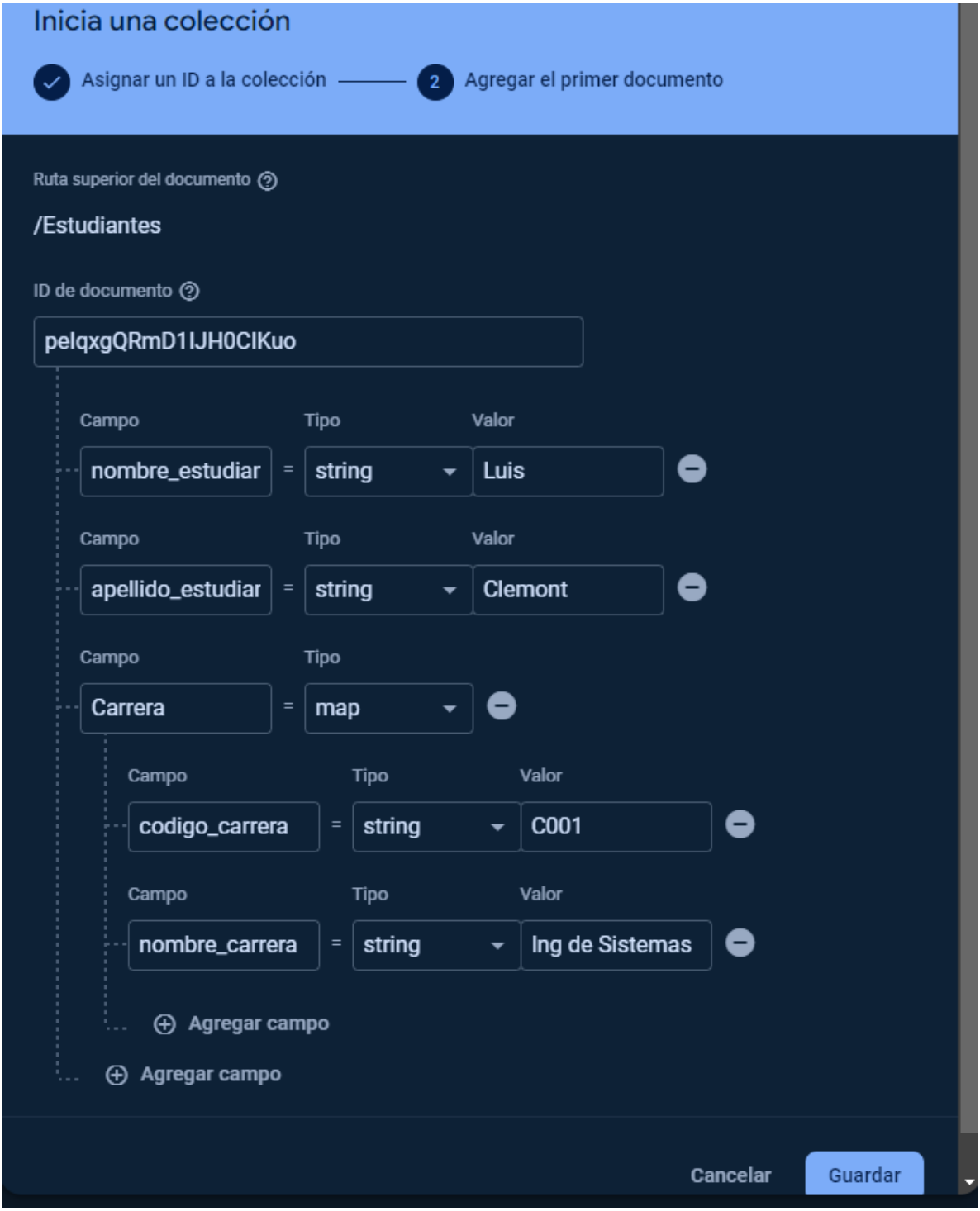

Inicia una colección
1
Asignar un ID a la colección
2
Agregar el primer documento
Ruta superior
/Carreras/sDp7xFvDXElLNMTkfL7B
ID de la colección
Materias
Cancelar
Siguiente


Inicia una colección
Asignar un ID a la colección
2
Agregar el primer documento
Ruta superior del documento
/Carreras/sDp7xFvDXElLNMTkfL7B/Materias
ID de documento
eGiqOcWUWm1LbFh445Yy
Campo
Tipo
Valor
nombre_materia
=
string
Base de datos
Campo
Tipo
Valor
nombre_profesor
=
string
Pedro
Campo
Tipo
Valor
apellido_profesor
=
string
Alvarez
Agregar campo
Cancelar
Guardar

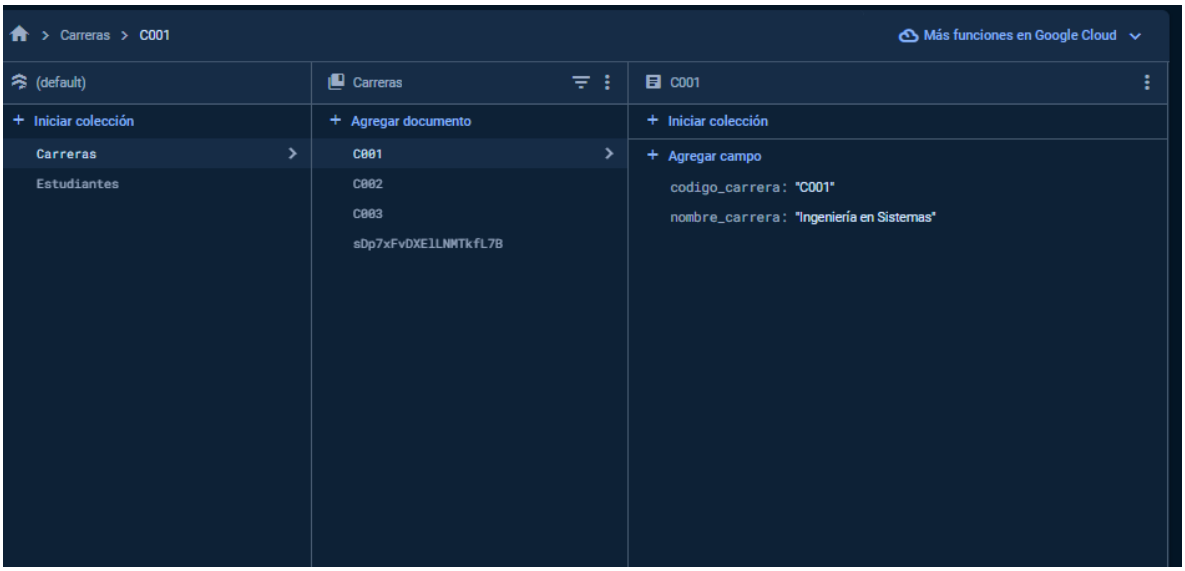

Inicia una colección
1 Asignar un ID a la colección
2 Agregar el primer documento
Ruta superior
/
ID de la colección
Estudiantes
Cancelar
Siguiente

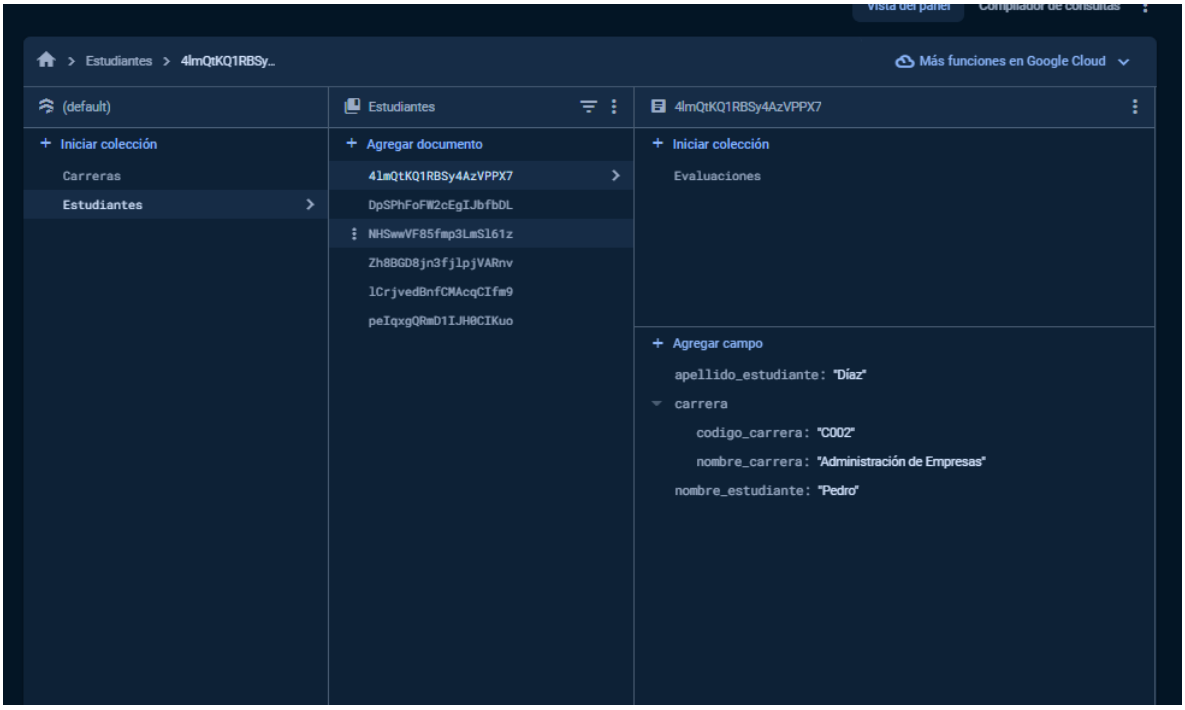

Inicia una colección
1 Asignar un ID a la colección
2 Agregar el primer documento
Ruta superior
/Estudiantes/pelqxgQRmD1IJH0CIKuo
ID de la colección
Evaluaciones
Cancelar
Siguiente


Inicia una colección
Asignar un ID a la colección
2
Agregar el primer documento
Ruta superior del documento
/Estudiantes/pelqxgQRmD1IJH0ClKuo/Evaluaciones
ID de documento
flh9yjHSyoFwGIhyPh31
Campo
Tipo
Valor
nombre_evaluaci
string
Examen 1
Campo
Tipo
fecha_evaluacior
timestamp
Fecha
18 abr 2024
Hora
08:45:00
Campo
Tipo
Valor
nota_evaluacion
number
20
Campo
Tipo
materia
map
Campo
Tipo
Valor
nombre_materia
string
Base de datos
Campo
Tipo
Valor
nombre_profesor
string
Pedro
Campo
Tipo
Valor
apellido_profesor
string
Alvarez
Agregar campo
Agregar campo


Inicia una colección
Asignar un ID a la colección
2
Agregar el primer documento
Ruta superior del documento
/Estudiantes
ID de documento
pelqxgQRmD1IJH0CIKuo
Campo
Tipo
Valor
nombre_estudiar
=
string
Luis
Campo
Tipo
Valor
apellido_estudiar
=
string
Clemont
Campo
Tipo
Carrera
=
map
Campo
Tipo
Valor
codigo_carrera
=
string
C001
Campo
Tipo
Valor
nombre_carrera
=
string
Ing de Sistemas
Agregar campo
Agregar campo
Cancelar
Guardar

Resultado final :



## ¿QUÉ BASE DE DATOS SERIA LA MEJOR OPCION PARA IMPLEMENTAR REACT NATIVE?

La mejor opción para implementar seria MySQL, porque es una base de datos escalable, fácil de implementar y soporta consultas complejas, además como esta publicada en un servidor en la nube pueden conectarse múltiples dispositivos a la vez; así mismo los costos de integración son relativamente mas bajos que la utilización de las bases de datos NoSQL.

## CONCLUSIÓN DE INVESTIGACION

El conocer más de una forma de llevar las bases de datos y la representación en los diferentes sistemas, es de gran utilidad en cualquier entorno de desarrollo, generando en el programador mejores competencias en el ámbito laboral como personal, debido a que proporciona mayores conocimientos útiles tanto en entornos personales como laborales.

## CONCLUSION DE IMPLEMENTACION

Base de datos utilizada: MySQL

Se utilizó MySQL por su fácil instalación y a su amplia documentación disponible. Es una opción popular para sistemas de gestión de bases de datos relacionales, lo que facilita tanto la implementación inicial como el mantenimiento continuo de las bases de datos.

La sintaxis SQL es intuitiva y fácil de entender, lo que permite desarrollar consultas de manera rápida y eficiente.

Apreciaciones finales bases de datos relacionales (SQL):

Reducción de la redundancia, la normalización permite evitar la redundancia de datos al dividir las tablas en entidades más pequeñas y relacionadas entre sí.

Facilita la actualización y modificación de los datos, ya que estos están distribuidos en tablas relacionadas de manera lógica.

Mejora el rendimiento de las consultas al permitir una distribución eficiente de los datos.

Ayuda a mantener la integridad de los datos al aplicar restricciones y reglas de integridad referencial.

Es importante evaluar las necesidades específicas del proyecto y las características de cada tecnología antes de elegir entre una base de datos SQL o NoSQL.

La normalización en bases de datos relacionales proporciona una estructura sólida y consistente para los datos, mientras que la desnormalización en bases de datos NoSQL puede mejorar el rendimiento y la accesibilidad de los datos, especialmente en entornos distribuidos y escalables, en el caso de una base de datos NoSQL es posible que se requiera un enfoque más flexible y orientado al caso de uso.

## BIBLIOGRAFÍA

FIREBASE. (20 de 03 de 2024). *Base de datos en tiempo real de Firebase*. Obtenido de https://firebase.google.com/docs/database?hl=es

FIREBASE. (22 de 03 de 2024). *Elija una base de datos: Cloud Firestore o Realtime Database*. Obtenido de https://firebase.google.com/docs/firestore/rtdb-vs-firestore?hl=es

FIREBASE. (22 de 03 de 2024). *Tienda de Fuegos en la nube*. Obtenido de https://firebase.google.com/docs/firestore?hl=es

UNIR-. (17 de 06 de 2021). *UNIVERSIDAD EN INTERNET*. Obtenido de NoSQL vs. SQL: características, diferencias y contextos de uso de estas tecnologías de SGBD: https://www.unir.net/ingenieria/revista/nosql-vs-sql/