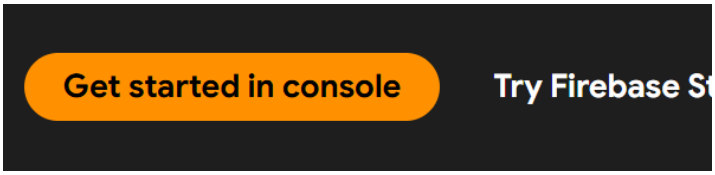
Firebase (II)

Vamos a ver cómo utilizamos una Base de Datos Documental con Firebase. Para ello, tendremos que acceder a:

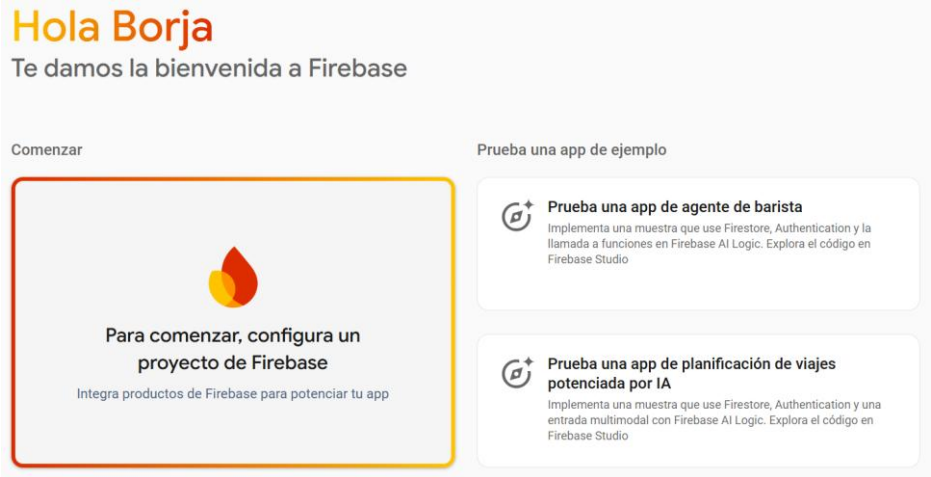
https://firebase.google.com/

Tendremos que usar nuestra cuente de Google para acceder a este servicio. Obviamente, es de **pago**; pero nos dan acceso limitado a ciertas partes.

Para acceder a Firebase, desde la página principal seleccionamos la opción **Get Started in Console**. Una nota antes de seguir. Han añadido opciones de IA (Gemini) a la plataforma, que en ocasiones es bastante molesta, sobre todo al acceder a la documentación que te ofrecen para buscar información. Pero es lo que hay.

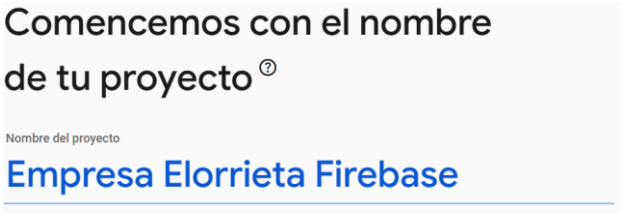


En la siguiente pantalla, tenemos acceso a la plataforma. A nivel elemental, Firebase se organiza en **Proyectos**. Cada uno de los proyectos, como veremos a continuación, nos ofrece una serie de recursos que podemos utilizar para nuestras aplicaciones. En la imagen inferior, ves las opciones de crear un nuevo proyecto. Cuando tengas alguno creado, los verás listados a tu derecha.

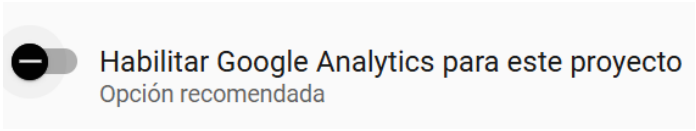


Nuevo Proyecto

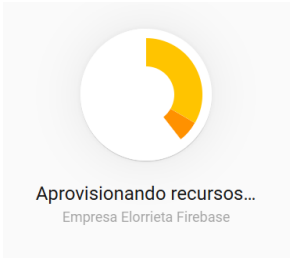
Para configurar un nuevo proyecto, vamos a seguir los siguientes pasos. Primero, le damos un nombre de proyecto representativo. En nuestro caso, **Empresa Elorrieta Firebase.**



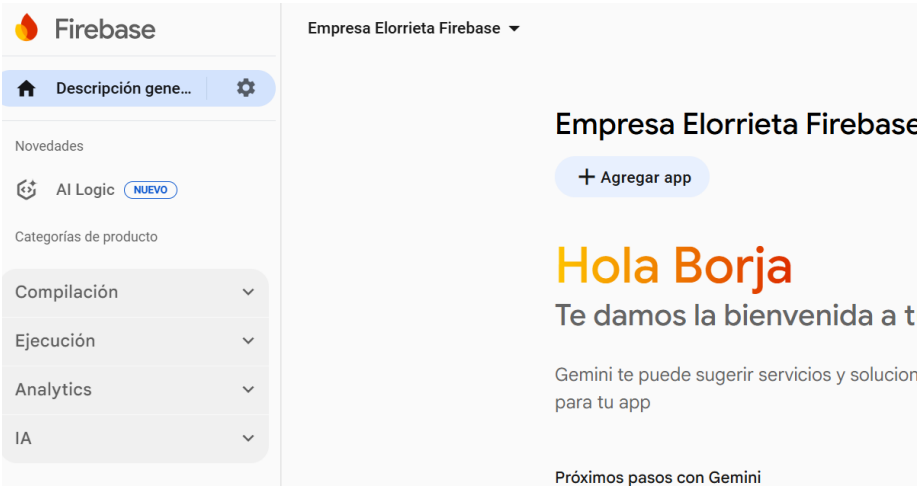
En la siguiente ventana, deshabilitaremos Google Analytics. No vamos

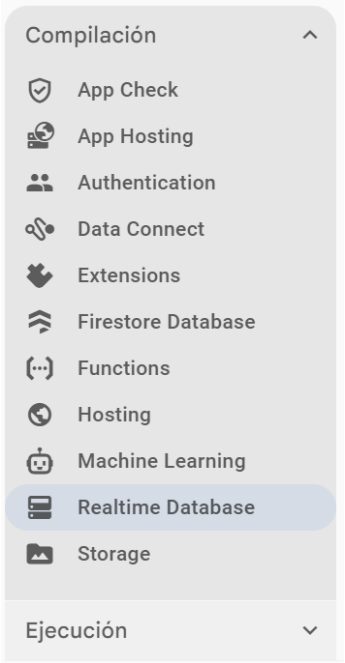


Al darle aceptar, se nos generará de forma automática el proyecto.



Y podremos acceder al panel principal del Proyecto.

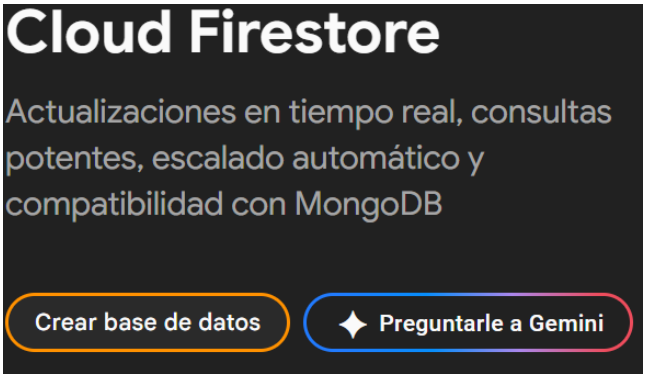




Creando la Base de Datos en Firestore

A la izquierda, se nos ofrece una serie de recursos distintos que podemos vincular a nuestro proyecto.

Seleccionamos la **Firestore Database**. Después **Crear Base de Datos**. Esto nos ofrece un nuevo menú de selección.

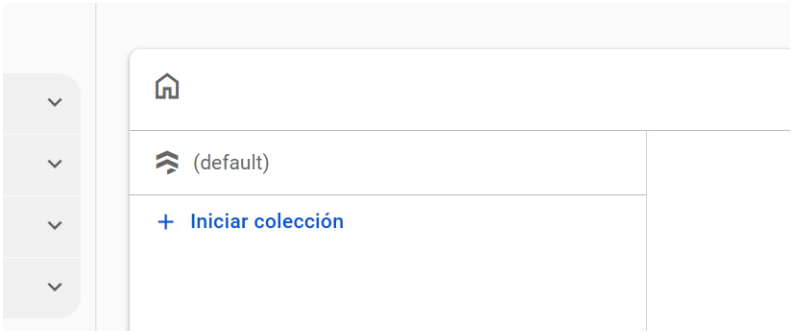


En el siguiente menú escogemos de forma sucesiva:

- **Edición Estándar**.
- **Ubicación**: la más cercana a ti que tengas en el combo.
- Comenzar en **modo de prueba**.

Para completar, le damos a crear.

Ya estamos listos para comenzar a trabajar con la Base de Datos. Para ello, pulsa **Iniciar Colección** y comienza a definirla.



Antes de continuar, detalles que tienes que recordar:

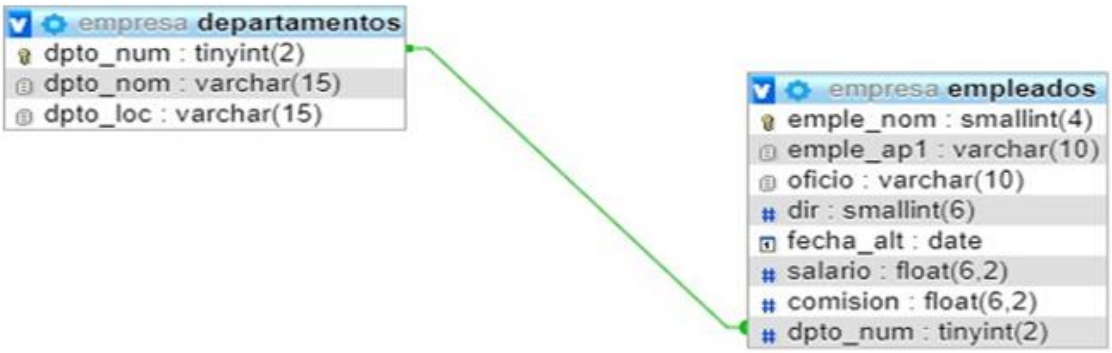
- Esto NO es una BBDD de SQL. Es Documental.
- Es muy flexible. Nada da error. Tienes control total. Puedes poner lo que quieras donde quieras. O meter cosas diferentes en un mismo sitio.
- Dispondrás de formas para acceder a las diferentes colecciones de Documentos. Y se usa JSON.

Mi Base de Datos

Para este ejemplo, vamos a crear una base de Datos de **Empresas**. Esta base de datos va a almacenar información referente a los departamentos y a los empleados de la empresa.

De cada departamento, queremos almacenar su número, su nombre, y su localización. De cada empleado, su nombre, empleo, oficio, dirección, fecha de alta, salario, comisión y a qué departamento pertenece.

Si esto fuese una BBDD de tipo Relacional (SQL) inmediatamente habrás llegado a la siguiente solución:



Y habrías generado lo siguiente en tu MySQL:

Departamentos

Base de datos: empresa » Tabla: departamentos

+ Opciones

	dpto_num	dpto_nom	dpto_loc
Editar Copiar Borrar	10	CONTABILIDAD	SEVILLA
Editar Copiar Borrar	20	INVESTIGACION	MADRID
Editar Copiar Borrar	30	VENTAS	BARCELONA
Editar Copiar Borrar	40	PRODUCCION	BILBAO

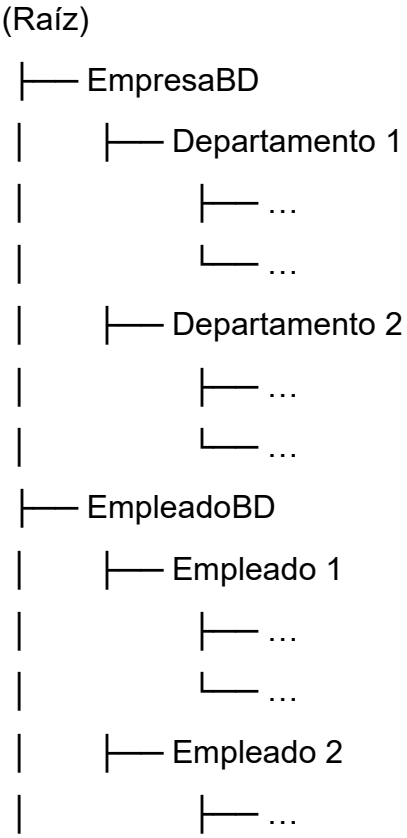
Empleados

Base de datos: empresa » Tabla: empleados

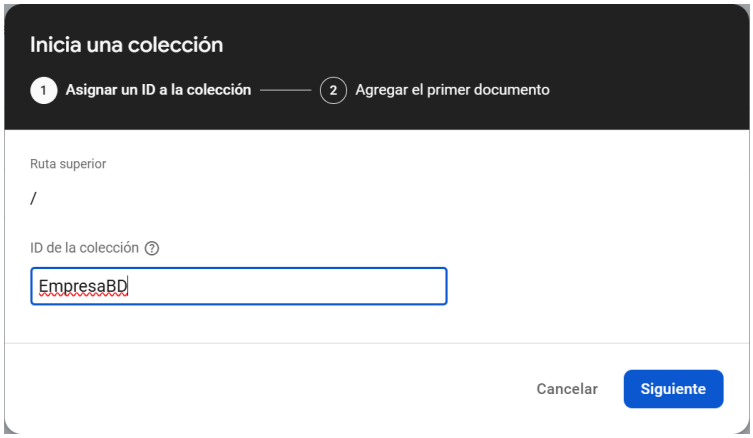
+ Opciones

	emple_nom	emple_ap1	oficio	dir	fecha_alt	salario	comision	dpto_num
Editar Copiar Borrar	7369	SANCHEZ	EMPLEADO	7902	1990-12-17	1040.00	NULL	20
Editar Copiar Borrar	7499	ARROYO	VENDEDOR	7698	1990-02-20	1500.00	390.00	30
Editar Copiar Borrar	7521	SALA	VENDEDOR	7698	1991-02-22	1625.00	650.00	30
Editar Copiar Borrar	7566	JIMENEZ	DIRECTOR	7839	1991-04-02	2900.00	NULL	20
Editar Copiar Borrar	7654	MARTIN	VENDEDOR	7698	1991-09-29	1600.00	1020.00	30
Editar Copiar Borrar	7698	NEGRO	DIRECTOR	7839	1991-05-01	3005.00	NULL	30
Editar Copiar Borrar	7782	CEREZO	DIRECTOR	7839	1991-06-09	2885.00	NULL	10
Editar Copiar Borrar	7788	GIL	ANALISTA	7566	1991-11-09	3000.00	NULL	20

Pero, sintiéndolo mucho, este planteamiento **NO SIRVE** para trabajar con Bases de Datos Documentales. Este tipo de BBDD todo tiene que ver con el manejo de grandes volúmenes de datos, por lo que se plantean soluciones diferentes. Obviamente, en este caso, vamos a tener colecciones de departamentos y colecciones de empleados. Cada departamento y cada empleado será un **documento**. Y para organizar la estructura, vamos a crear un sencillo árbol de forma similar a esta.

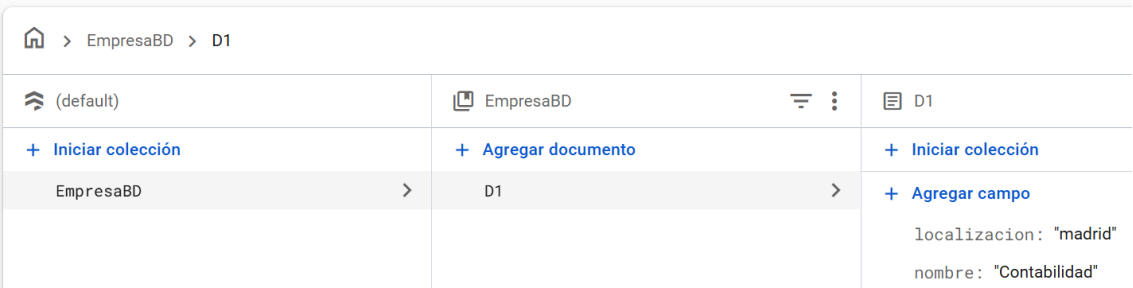


Por tanto, vamos a **FireBase** y seleccionamos **iniciar colección**.

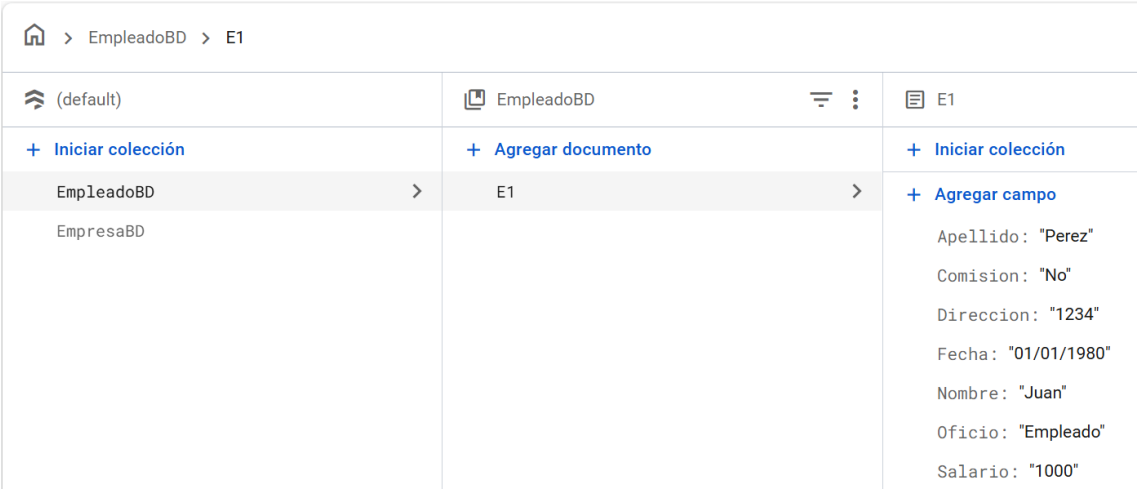


FireBase no permite que existan colecciones vacías. Por tanto, siempre que creemos una colección, será necesario añadirle al menos un Documento.

Nótese que no añadimos el campo **dpto_num** de la tabla departamentos porque es una clave primaria generada para SQL, innecesaria en nuestra base de datos documental. El resultado será

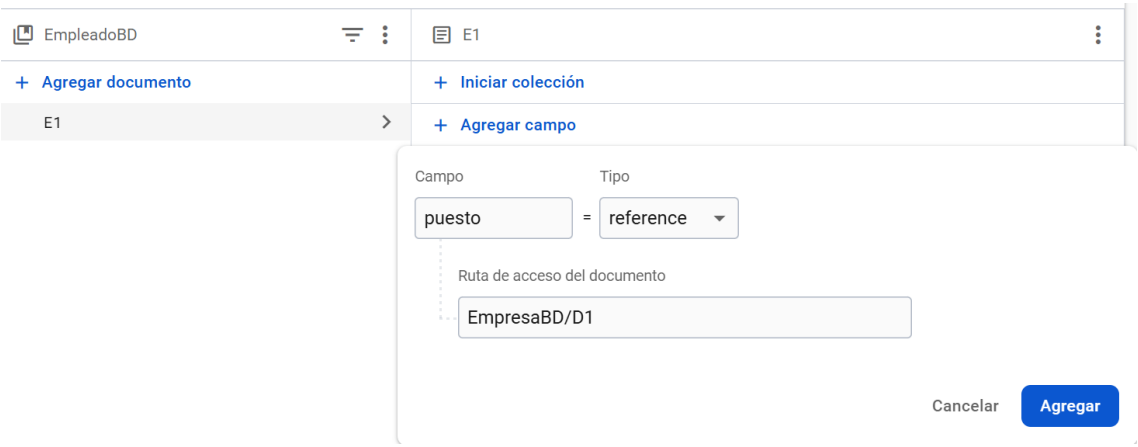


Siguiendo la misma lógica, vamos a generar una nueva colección para los empleados:



Con esto, ya tendríamos preparada la Base de Datos. Sólo nos queda un pequeño detalla… ¿cómo relacionar un Documento de Empleado con un Documento de Empresa? Esa relación 1:N es, al fin y al cabo, información. Bien, pues… puedes hacerlo como quieras. No obstante, la forma más simple, es mediante una **referencia**. Esto NO es lo mismo que una relación SQL. Es, simplemente, un campo que te lleva a otro documento.

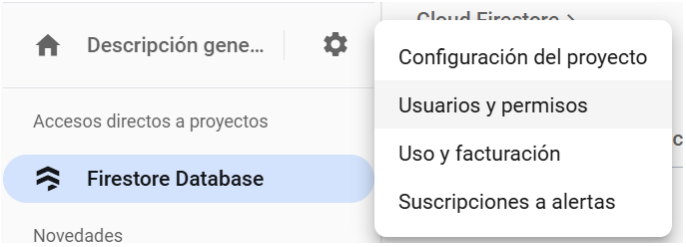
En nuestro caso, podemos hacer:



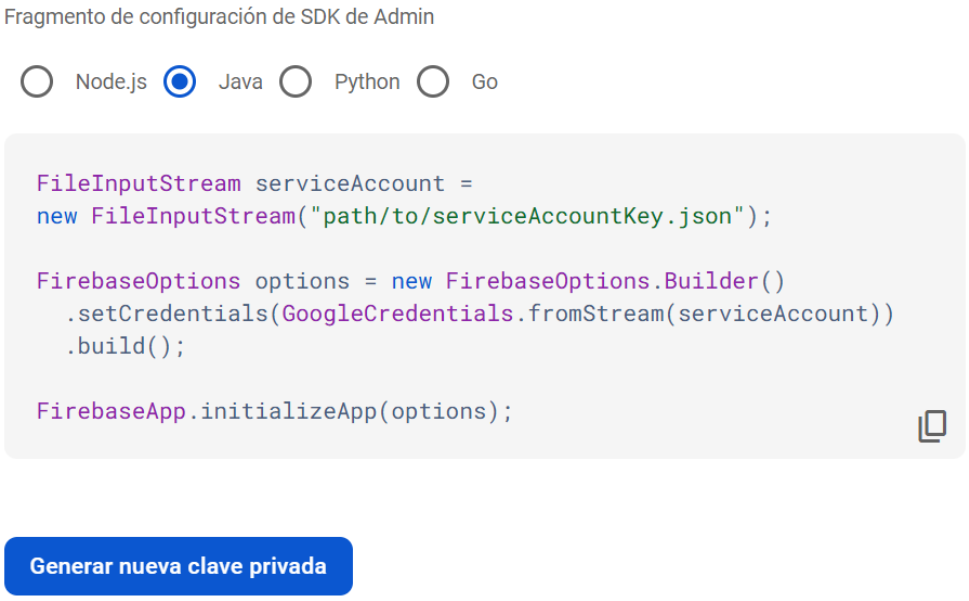
Vemos ahora cómo dar un uso a esta base de datos desde Java.

Accediendo desde Java

Antes de programar una sola línea de código, es necesario crear los permisos para acceder a la BBDD desde terceras aplicaciones. Vamos a la **Descripción General del Proyecto** y le damos a **Usuarios y Permisos**.



En la pestaña de **Cuentas de Servicio** podemos configurar una cuenta que vamos a usar para autenticarnos en Firebase. Es equivalente un certificado que nos da permisos de acceso a nuestro proyecto. Pulsando **Java**, nos genera automáticamente el código que tenemos que usar (un poco más adelante). Lo que nos interesa aquí es darle a **Generar nueva clave privada**.



En mi caso, esto me generó un fichero json llamado:

empresa-elorrieta-firebase-firebase-adminsdk-fbsvc-aec97a7458.json

Por sencillez, voy a renombrarlo a **"EmpresaBD.json"**. Este fichero tendrá que estar en mi proyecto Java y tendré que referenciarlo desde el código que vemos aquí arriba.

NOTA: Mira mejor el ejemplo solucionado. No te fíes del código Java este. Hay demasiados detalles que ir solucionando sobre la marcha. Trust me.

Ahora, ya podemos comenzar a trabajar con Java. Los pasos son sencillos:

1) Crea un proyecto Maven
2) Añade las dependencias del Firebase y el Gson.
3) Codifica un ejemplo.

A modo de ejemplo, vamos a recuperar todos los documentos Empresa. Para ello, creamos una clase específica: **OperacionesFirebase**.

```
public OperacionesFirebase() throws FireBaseException {
    try {
        if (FirebaseApp.getApps().isEmpty()) {
            InputStream serviceAccount = OperacionesFirebase.class.getResourceAsStream(CREDENTIALS);
            FirebaseOptions options = FirebaseOptions.builder()
                    .setCredentials(GoogleCredentials.fromStream(serviceAccount)).build();
            FirebaseApp.initializeApp(options);
        }
    } catch (Exception e) {
        throw new FireBaseException("Error - " + e.getLocalizedMessage());
    }
}

@Override
public List<Empresa> getEmpresas() throws FireBaseException {
    List<Empresa> ret = null;

    try {

        Firestore dataBase = FirestoreClient.getFirestore();

        // Query...
        ApiFuture<QuerySnapshot> query = dataBase.collection(COLLECTION_EMPRESA).get();

        // Procesamos la query...
        QuerySnapshot querySnapshot = query.get();
        List<QueryDocumentSnapshot> empresas = querySnapshot.getDocuments();
        for (QueryDocumentSnapshot empresa : empresas) {
            ret = null == ret ? new ArrayList<Empresa>() : ret;
            ret.add(new Empresa(empresa.getId(), empresa.getString("nombre"),
                empresa.getString("localizacion")));
        }

    } catch (Exception e) {
        throw new FireBaseException("Error - " + e.getLocalizedMessage());
    }
    return ret;
}
```

El resultado es:

```
Empresa [id=D1, nombre=Contabilidad, localizacion=Madrid]
Empresa [id=D2, nombre=Produccion, localizacion=Bilbao]
```

Volviendo con el tema de las **referencias**, también es posible usar la definida en el Empleado Juan para recuperar su Empresa:

```
public Empleado getEmpleado(String name) throws FireBaseException {
    Empleado ret = null;
    try {

        Firestore dataBase = FirestoreClient.getFirestore();

        // Query... para buscar Empleado
        ApiFuture<QuerySnapshot> query = dataBase.collection(COLLECTION_EMPLEADO)
                .whereEqualTo("Nombre", name)
                .get();

        // Procesamos la query...
        QuerySnapshot querySnapshot = query.get();
        List<QueryDocumentSnapshot> empleados = querySnapshot.getDocuments();
        for (QueryDocumentSnapshot empleado : empleados) {

            ret = new Empleado(empleado.getString("Nombre"), empleado.getString("Apellido"),
                    empleado.getString("Oficio"),
                    empleado.getString("Direccion"),
                    empleado.getString("Fecha"),
                    empleado.getString("Salario"),
                    empleado.getString("Comision"), null);

            // Nos falta los datos de la Empresa. Es una referencia que viene con el campo
            // "puesto".
            DocumentReference puestoRef = (DocumentReference) empleado.getData().get("puesto");
            ret.setEmpresa(puestoRef == null ? null : getEmpresaByID(puestoRef.getId()));
            break;
        }

    } catch (Exception e) {
        throw new FireBaseException("Error - " + e.getLocalizedMessage());
    }
    return ret;
}
```

Una vez obtenida la ID:

```
@Override
public Empresa getEmpresaByID(String id) throws FireBaseException {
    Empresa ret = null;
    try {

        Firestore dataBase = FirestoreClient.getFirestore();

        // Query...
        DocumentReference documentReference = dataBase.collection(COLLECTION_EMPRESA).document(id);
        DocumentSnapshot documentSnapshot = documentReference.get().get();

        // Procesamos ...
        if (null != documentSnapshot) {
            ret = new Empresa(documentSnapshot.getId(), documentSnapshot.getString("nombre"),
                    documentSnapshot.getString("localizacion"));
        }
    } catch (Exception e) {
        throw new FireBaseException("Error - " + e.getLocalizedMessage());
    }
    return ret;
}
```

El resultado es:

```
Empleado [nombre=Juan, apellido=Perez, oficio=Empleado, direccion=1
|---Empresa [id=D1, nombre=Contabilidad, localizacion=Madrid]
```

Otras consultas

Existe una gran variedad de consultas que puedes realizar:

- Obtener Empresa Contabilidad:

```
db.collection("empresaBD").whereEqualTo("nombre", "CONTABILIDAD").get();
```

- Obtener Empleados con salario menor a 3000:

```
empleadosCol.whereLessThan("salario", 3000).get();
```

- Obtener empleados con salario menor a 3000 ordenado por apellido:

```
empleadosCol.whereLessThan("salario", 3000).orderBy("apellido").get();
```

- Obtener empleados con salario menor a 3000 y oficio igual a DIRECTOR ordenado por apellido:

```
empleadosCol.whereLessThan("salario", 3000).whereEqualTo("oficio",
"DIRECTOR").orderBy("apellido").get();
```

Errores típicos

A la hora de trabajar con Firebase, unos cuantos consejos:

- Es posible que tengas que generar de nuevo el fichero de credenciales. No desesperes. Es porque estamos de prueba. Es lo que hay.

- Firebase no te da Exception como el MySQL si metes una query errónea. Puede ser que te de nulo, porque no ha encontrado nada. Fíjate en los nombres de los campos, si has puesto mayúsculas, etc.

- Y recuerda: a una Base Documental no le importa si en una lista de Manzanas metes Peras, o si hay una Pera radicalmente distinta a otra. Gestionar eso es cosa del programador.