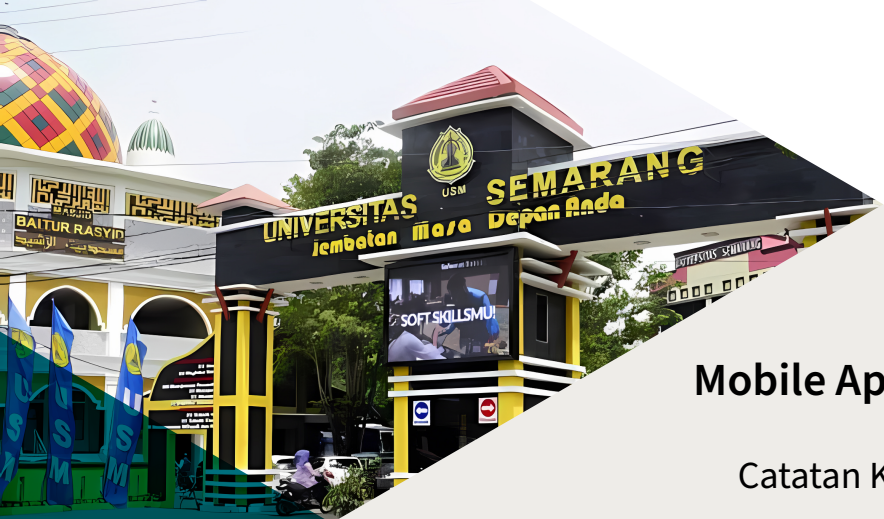


Mobile Application

Catatan Kuliah #6

Alauddin Maulana Hirzan, M. Kom
0607069401

Collections dan Documents

Collections dan Documents

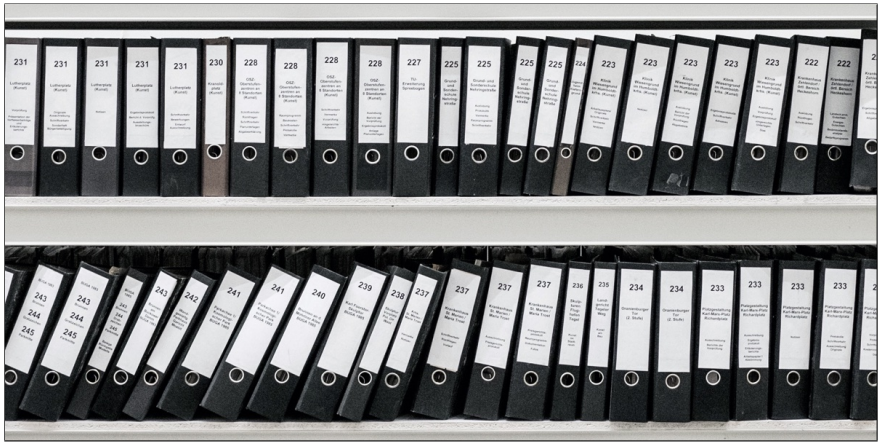
Model Data Koleksi dan Dokumen #1

Firebase dan beberapa DBMS lainnya menerapkan sistem collections sebagai model penyimpanan datanya. Berbeda dengan konsep biasanya yang menggunakan baris dan kolom, model ini meniru teknik dokumen yang biasa digunakan sehari-hari

Model ini sudah diterapkan bertahun-tahun ketika sebuah kantor melakukan pengarsipan dokumen.

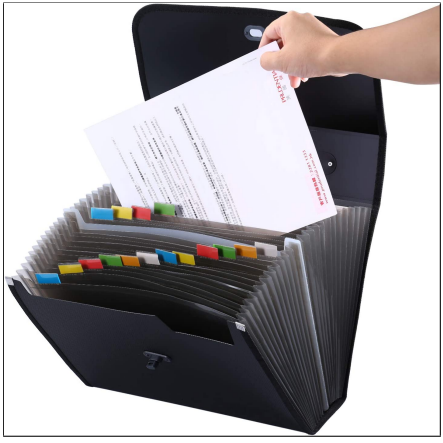
Collections dan Documents

Model Data Koleksi dan Dokumen #2



Collections dan Documents

Model Data Koleksi dan Dokumen #3



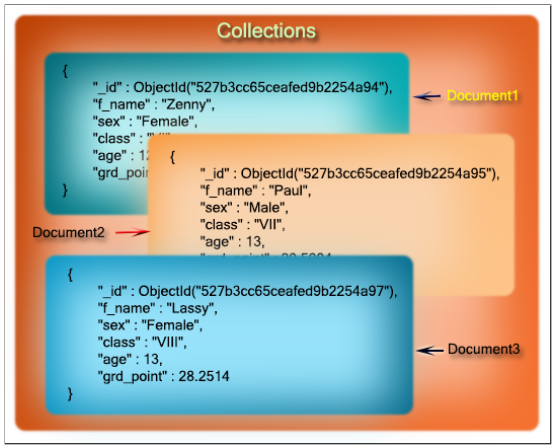
Collections dan Documents

Model Data Koleksi dan Dokumen #4

- **Dokumen** adalah unit penyimpanan data dalam database NoSQL. Data disimpan dalam bentuk JSON di dalam dokumen tersebut.
- **Koleksi** dapat menyimpan sejumlah dokumen. Suatu koleksi dapat menyimpan dokumen-dokumen yang strukturnya tidak sama. Sedangkan, dalam database relasional seperti MySQL, skema mendefinisikan organisasi / struktur data dalam database.

Collections dan Documents

Model Data Koleksi dan Dokumen #5



Batasan Model Data

Batasan Model Data

Batasan Nama

Agar nama sebuah dokumen maupun koleksi sesuai dengan spesifikasi standar database, ada beberapa persyaratan yang harus dipenuhi berupa:

- Nama koleksi harus dimulai dengan huruf atau garis bawah.
- Nama Koleksi dapat berisi angka.
- Tidak dapat menggunakan karakter "$" dalam nama koleksi. "$" telah dipesan.
- Nama Koleksi tidak boleh melebihi 128 karakter

Batasan Model Data

Batasan Standar #1

- Kendala pada ID koleksi
  - Harus berupa karakter UTF-8 yang valid
  - Tidak boleh lebih dari 1.500 byte
  - Tidak boleh berisi garis miring (/)
  - Tidak boleh hanya terdiri dari titik tunggal (.) atau titik ganda (..)
  - Tidak dapat mencocokkan ekspresi reguler __.*__
- Kedalaman maksimum subkoleksi
  - 100

Batasan Model Data

Batasan Standar #2

- Batasan pada ID dokumen
  - Harus berupa karakter UTF-8 yang valid
  - Tidak boleh lebih dari 1.500 byte
  - Tidak boleh berisi garis miring (/)
  - Tidak boleh hanya terdiri dari titik tunggal (.) atau titik ganda (..)
  - Tidak dapat mencocokkan ekspresi reguler __.*__
  - Jika Anda mengimpor entitas Datastore ke dalam database Firestore, ID entitas numerik ditampilkan sebagai __id[0-9]+__

Batasan Model Data

Batasan Standar #3

- Ukuran maksimum untuk nama dokumen
  - 6 KiB
- Ukuran maksimum untuk dokumen
  - 1 MiB (1.048.576 byte)
- Batasan pada nama bidang
  - Harus berupa karakter UTF-8 yang valid
- Ukuran maksimum nama bidang (field)
  - 1.500 byte

Teknik Kueri Database

Teknik Kueri Database

Akses Data #1 - Set

Untuk melakukan pengambilan data menggunakan **Kotlin** baik Desktop maupun Mobile bisa menggunakan kueri sederhana seperti berikut :

- val cities = db.collection("cities")

Variabel **cities** ini akan berisikan daftar dokumen-dokumen yang ada di dalam koleksi **cities** (Jika ada). Jika kosong, maka koleksi adalah **array** / **dictionary** yang kosong.

Teknik Kueri Database

Akses Data #2 - Set

Untuk membuat dokumen baru bisa dilakukan dengan cara yang mudah, namun sebelumnya wajib membuat data sederhana:

- val data1 = hashMapOf("kota" to "Semarang", "provinsi" to "Jawa Tengah")

Data ini jika dilihat secara **dictionary** akan terlihat sebagai berikut :
["nama":"Semarang","provinsi":"Jawa Tengah"]

Teknik Kueri Database

Akses Data #3 - Set

Untuk menyimpan data tersebut ke sebuah dokumen dengan nama **Kota Semarang**, maka hanya perlu menggunakan perintah sederhana seperti berikut:

- cities.document("Kota Semarang").set(data1)

Dengan menggunakan kata kunci **set()**, maka **data1** akan dituliskan di dalam dokumen dengan nama **Kota Semarang** yang terletak di dalam koleksi **cities**. Kueri juga bisa dilakukan versi panjang

- db.collection("cities").document("Kota Semarang").set(data1)

Teknik Kueri Database

Akses Data #4 - Get

Pengambilan data bisa dilakukan dengan dua cara:

1. Iteratif / **one-by-one**
2. Langsung

Cara pertama saat ingin mengakses semua data satu per satu, sehingga dapat mengecek tiap kolom data yang ada di masing-masing dokumen. Sedangkan cara kedua cocok ketika sudah mengetahui koleksi dan dokumen tujuan.

Teknik Kueri Database

Akses Data #5 - Get - Iteratif

Teknik iteratif ini mengandalkan **FOR** atau **FOREACH** Loop untuk mengakses tiap elemen yang ada di Database.

Jika diketahui database dengan 5 koleksi dan 10 dokumen di dalamnya, maka untuk mengakses ini secara iteratif bisa dilakukan dengan cara yang mudah.

Contoh kode dalam Python:

```
for doc_key in db.keys():
    for document in db.document(doc_key):
        print(document.get())
```

Teknik Kueri Database

Akses Data #6 - Get - Iteratif

Cara membaca :

- Dengan menggunakan kode tersebut maka program akan mengambil **Kunci** (Daftar Koleksi) yang ada di dalam database.
- **Kunci/Nama Koleksi** akan digunakan untuk mengambil **Dokumen-Dokumen** yang ada di **Koleksi** tersebut
- Lalu tampilkan data yang ada di masing-masing dokumen.

Teknik Kueri Database

Akses Data #6 - Get - Langsung

Cara ini merupakan cara yang mudah, dan biasanya digunakan setelah **Koleksi** dan **Dokumen** sudah diketahui. Sebagai contoh mengakses data user setelah user itu login ke dalam aplikasi mobile.

Perhatikan 2 contoh kode berikut

1. ```
   data_semarang = db.collection("cities").document("Kota Semarang").get()
   ```
2. ```
   koleksi_kota = db.collection("cities")
   dokumen_semarang = koleksi_kota.document("Kota Semarang")
   data_semarang = dokumen_semarang.get()
   ```

Teknik Kueri Database

Akses Data #7 - Event Listener

Ketika melakukan kueri baik **Pengambilan** maupun **Penulisan** data, programmer dapat menempelkan sebuah **Event Listener** yang mampu mendeteksi **Keberhasilan** maupun **Kegagalan** Kueri.

Perhatikan kode berikut:

- db.collection("users").get()
- .addOnSuccessListener result -> print(result)
- .addOnFailureListener exception -> print(exception)

Offline Mode

Offline Mode

Offline Mode #1

Saat programmer menginisialisasi Cloud Firestore, programmer dapat mengaktifkan atau menonaktifkan persistensi offline. Untuk platform Android dan Apple, persistensi offline diaktifkan secara default. Untuk menonaktifkan persistensi, setel opsi PersistenceEnabled ke false.

Saat persistensi diaktifkan, Cloud Firestore menyimpan cache setiap dokumen yang diterima dari backend untuk akses offline. Cloud Firestore menetapkan ambang batas default untuk ukuran cache. Setelah melebihi default, Cloud Firestore secara berkala mencoba membersihkan dokumen lama yang tidak digunakan.

Offline Mode

Offline Mode #2

Saat perangkat sedang offline, jika pemrogram telah mengaktifkan kegigihan offline, pendengar akan menerima peristiwa mendengarkan saat data yang di-cache secara lokal berubah. Programmer dapat mendengarkan dokumen, koleksi, dan kueri.

Untuk memeriksa apakah pemrogram menerima data dari server atau cache, gunakan properti fromCache pada SnapshotMetadata dalam peristiwa snapshot Anda. Jika fromCache benar, data berasal dari cache dan mungkin basi atau tidak lengkap. Jika fromCache salah, data lengkap dan terkini dengan pembaruan terbaru di server.

Offline Mode

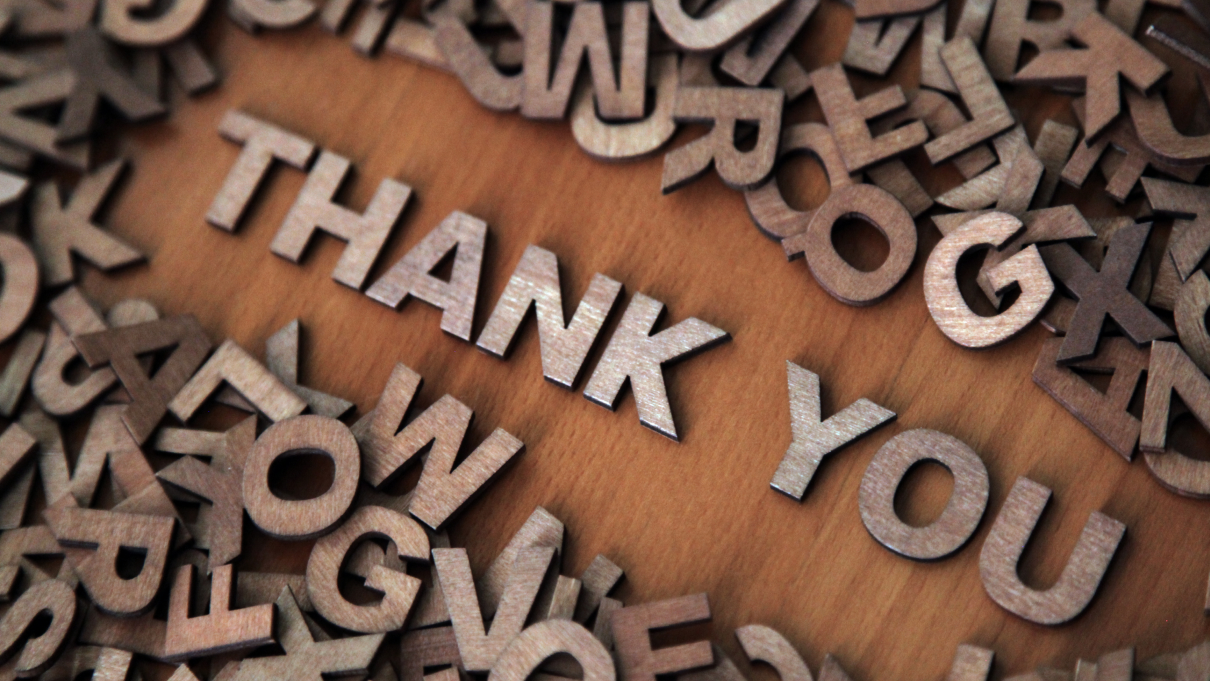
Offline Mode #2

Untuk mengeset perangkat ke mode Offline dapat dilakukan dengan mudah melalui perintah:

- db.disableNetwork().addOnCompleteListener { ... }

Lalu untuk mengembalikan mode Online:

- db.enableNetwork().addOnCompleteListener { ... }


THANK YOU