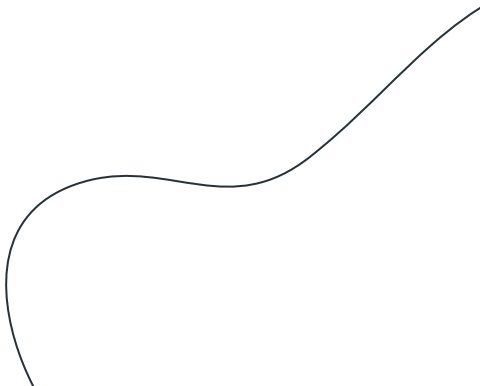
Database (Room)



Pertemuan 10
Mata Kuliah:
Pemrograman Aplikasi Mobile

Dosen:
Mahisa Ajy Kusuma, S.Kom., MMSI.

Room Database



Mengapa Aplikasi Membutuhkan Database?

Bayangkan aplikasi yang sudah kita buat.

- **Tanpa database.**
  Data hanya berada di variabel. Saat aplikasi ditutup, data dapat hilang.
- **Dengan database.**
  Data disimpan di perangkat dan dapat dibaca kembali saat aplikasi dibuka.

Database membantu aplikasi menyimpan data secara terstruktur dan persisten.

Room dan SQLite: Apa Hubungannya?

SQLite

- Database relasional yang berjalan di perangkat.
- Data disimpan dalam tabel dan baris.
- Android menyediakan dukungan SQLite.
- Developer dapat menulis SQL secara langsung.

Room

- Library bagian dari Android Jetpack.
- Bekerja di atas SQLite.
- Mengurangi kode boilerplate.
- Memvalidasi query SQL saat compile time.
- Mengubah hasil query menjadi objek Kotlin.

Room bukan pengganti mesin database SQLite.
Room adalah lapisan abstraksi di atas SQLite.

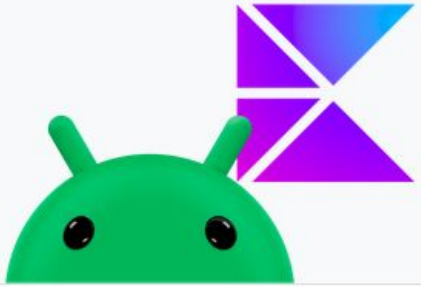
Save data in a local database using Room

Part of Android Jetpack.

Try with Kotlin Multiplatform

Kotlin Multiplatform allows sharing the database layer with other platforms. Learn how to set up and work with Room Database in KMP

Set up Room Database for KMP →



Apps that handle non-trivial amounts of structured data can benefit greatly from persisting that data locally. The most common use case is to cache relevant pieces of data so that when the device can't access the network, your users can still browse that content while they're offline.

The Room persistence library provides an abstraction layer over SQLite to allow fluent database access while harnessing the full power of SQLite. In particular, Room provides the following benefits:

- Compile-time verification of SQL queries.
- Convenience annotations that minimize repetitive and error-prone boilerplate code.
- Streamlined database migration paths.

We recommend using Room instead of using the SQLite APIs directly.

3 Komponen Utama Room

ENTITY

Bentuk / struktur data

DAO

Operasi dan query database

DATABASE

Penghubung dan konfigurasi

1. Entity: Mendefinisikan Tabel

Entity merepresentasikan sebuah tabel di database.

```kotlin
@Entity(tableName = "contacts")
data class Contact(
    @PrimaryKey(autoGenerate = true)
    val id: Int = 0,

    val name: String,
    val phone: String,
    val email: String
)
```

Penjelasan

@Entity
Menandai class sebagai tabel.

@PrimaryKey
Menentukan primary key.

autoGenerate = true
ID dibuat otomatis oleh database.

Setiap properti menjadi kolom.

2. DAO: Mendefinisikan Operasi Database

DAO (Data Access Object) berisi fungsi untuk membaca dan mengubah data.

```
@Dao
interface ContactDao {
    @Insert
    suspend fun insert(contact: Contact)

    @Query("SELECT * FROM contacts")
    fun getAll(): Flow<List<Contact>>

    @Update
    suspend fun update(contact: Contact)

    @Delete
    suspend fun delete(contact: Contact)
}
```

Anotasi penting

@Insert — menambah data

@Query — menjalankan SQL

@Update — mengubah data

@Delete — menghapus data

suspend / Flow mendukung operasi asynchronous dan observasi data.

3. Database: Menghubungkan Komponen

```
@Database(
    entities = [Contact::class],
    version = 1,
    exportSchema = false
)
abstract class AppDatabase : RoomDatabase() {
    abstract fun contactDao(): ContactDao
}
```

Tugas Database

- Mendaftarkan Entity.
- Menentukan versi database.
- Menyediakan akses ke DAO.
- Menjadi pintu masuk aplikasi ke Room.

libs.versions.toml

```
[versions]
agp = "9.5.1"
coreKtx = "1.19.1"
junit = "4.13.2"
junitVersion = "1.3.0"
espressoCore = "3.7.0"
appcompat = "1.8.0"
material = "1.14.0"
activityKtx = "1.13.0"
constraintlayout = "2.2.2"
room = "2.8.5"
ksp = "2.3.13"

[libraries]
androidx-core-ktx = { group = "androidx.core", name = "core-ktx", version.ref = "coreKtx" }
junit = { group = "junit", name = "junit", version.ref = "junit" }
androidx-junit = { group = "androidx.test.ext", name = "junit", version.ref = "junitVersion" }
androidx-espresso-core = { group = "androidx.test.espresso", name = "espresso-core", version.ref = "espressoCore" }
androidx-appcompat = { group = "androidx.appcompat", name = "appcompat", version.ref = "appcompat" }
material = { group = "com.google.android.material", name = "material", version.ref = "material" }
androidx-activity-ktx = { group = "androidx.activity", name = "activity-ktx", version.ref = "activityKtx" }
androidx-constraintlayout = { group = "androidx.constraintlayout", name = "constraintlayout", version.ref = "constraintlayout" }
androidx-room-runtime = { group = "androidx.room", name = "room-runtime", version.ref = "room" }
androidx-room-ktx = { group = "androidx.room", name = "room-ktx", version.ref = "room" }
androidx-room-compiler = { group = "androidx.room", name = "room-compiler", version.ref = "room" }

[plugins]
android-application = { id = "com.android.application", version.ref = "agp" }
ksp = { id = "com.google.devtools.ksp", version.ref = "ksp" }
```

build.gradle.kts (:app)

You can use the Project Structure dialog to view and edit your project configuration

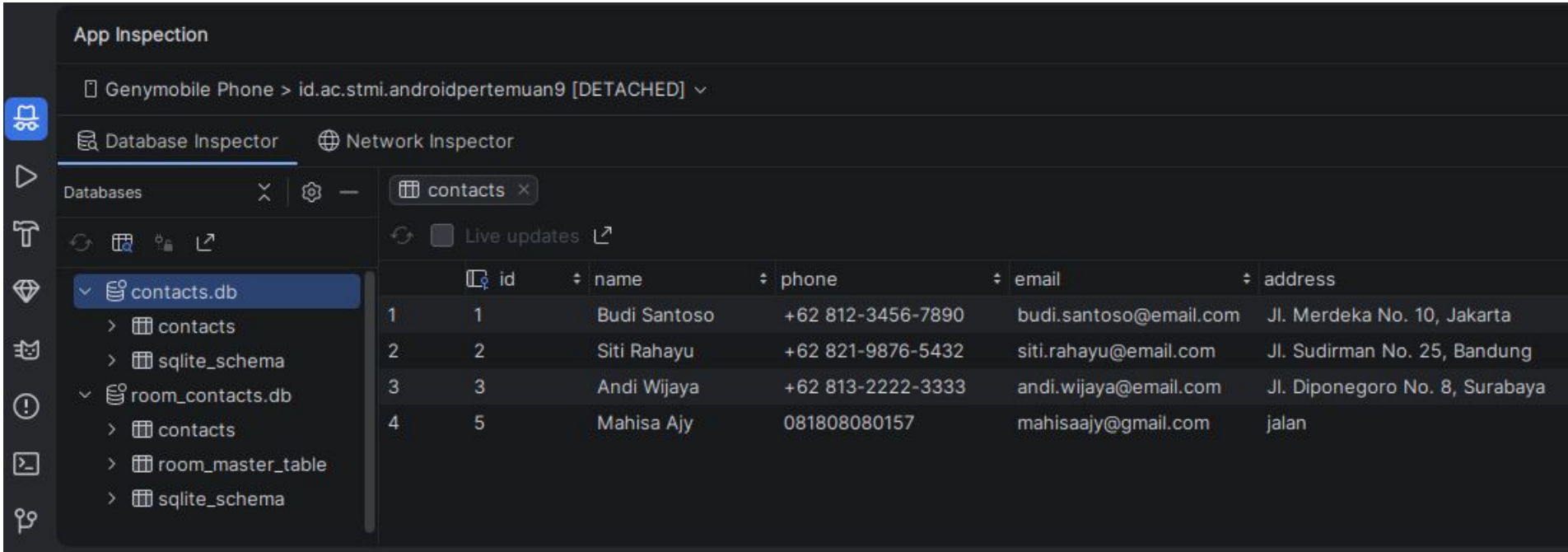
```
plugins {
    alias(libs.plugins.android.application) com.android.application:9.5.1
    alias(libs.plugins.ksp) com.google.devtools.ksp:2.3.13

}
```

```
dependencies {
    implementation(libs.androidx.activity.ktx) androidx.activity:activity-ktx:1.13.0
    implementation(libs.androidx.appcompat) androidx.appcompat:appcompat:1.8.0
    implementation(libs.androidx.constraintlayout) androidx.constraintlayout:constraintlayout:2.2.2
    implementation(libs.androidx.core.ktx) androidx.core:core-ktx:1.19.1
    implementation(libs.material) com.google.android.material:material:1.14.0
    testImplementation(libs.junit) junit:junit:4.13.2
    androidTestImplementation(libs.androidx.espresso.core) androidx.test.espresso:espresso-core:3.7.0
    androidTestImplementation(libs.androidx.junit) androidx.test.ext:junit:1.3.0
    implementation(libs.androidx.room.runtime) androidx.room:room-runtime:2.8.5
    implementation(libs.androidx.room.ktx) androidx.room:room-ktx:2.8.5
    ksp(libs.androidx.room.compiler) androidx.room:room-compiler:2.8.5
}
```

Mengakses Database

File → View → Tool Windows → App Inspection (Tab Database Inspector)



PRAKTIK

Tugas Praktikum

Membuat Aplikasi Android dengan menggunakan Room Database. Minimal terdapat 2 tabel dalam Aplikasi tersebut. Berikut ini hanya contoh:

1. Ide aplikasi



1. Contact Manager

Mengelola kontak dan nomor telepon.

- Tabel `Contact`
- Tabel `PhoneNumber`
- Satu kontak bisa memiliki beberapa nomor telepon.

Mudah



2. Library Management

Aplikasi pengelolaan buku perpustakaan.

- Tabel `Category`
- Tabel `Book`
- Satu kategori memiliki banyak buku.

Mudah



3. Expense Tracker

Aplikasi pencatatan pengeluaran.

- Tabel `Category`
- Tabel `Transaction`
- Satu kategori memiliki banyak transaksi.

Mudah



4. Student Course

Aplikasi pengelolaan mahasiswa dan mata kuliah.

- Tabel `Student`
- Tabel `Course`
- Bisa dikembangkan menjadi relasi many-to-many menggunakan tabel `StudentCourse`.

Menengah



5. Recipe Book

Aplikasi penyimpanan resep masakan.

- Tabel `Recipe`
- Tabel `Ingredient`
- Satu resep memiliki banyak bahan.

Mudah



6. Task Manager

Aplikasi pengelolaan proyek dan tugas.

- Tabel `Project`
- Tabel `Task`
- Satu proyek memiliki banyak tugas.

Mudah



Thanks!

Do you have any questions?
mahisaajy@gmail.com
+62 81-808080-157
https://mahisaajy.web.id

CREDITS: This presentation template was created by **Slidesgo**, including icons by **Flaticon**, infographics & images by **Freepik** and illustrations by **Stories**

Please keep this slide for attribution