Discussion-3 Kotlin Compare and contrast SQLite and Room Persistence Library

Discussion Topic:

Please choose one of the following questions to discuss in your initial post:

Compare and contrast SQLite and Room Persistence Library in terms of usage and features. Which one would you prefer for your Android applications and why?
Discuss the importance of data persistence in mobile applications. What are the different approaches to storing data locally in Android?
Share your implementation of the note-taking application assignment. Explain any design choices you made and any difficulties you faced while implementing the database operations.

My Post:

Hello Class,

Data persistence is integral to building functional and robust applications within the Android ecosystem. Therefore, choosing the right data persistent solution for a specific application is essential, as it can significantly influence an app's performance, maintainability, and, consequently, the user experience. This article explores SQLite database and the Room Persistence Library.

Data Persistence in Mobile Applications

Data persistence in mobile Applications can be defined as the capability of an application to store data locally with the goal of being retrieved even after the program is terminated and restarted (MongoDB, n.d.; Cuello, 2023). This is essential for preventing data losses, remembering the application state after the user left the application (e.g. in video game user progression), for offline functionality, and better performance if the data is retrieved locally rather than from a server. In other words, data persistence is essential for building functional and robust Android applications that provide a seamless user experience.

Storing Data Locally in Android

When storing data locally, Android offers several options. Storage options like sharing preferences that use share key-values pairs, internal storage that stores files on-device, external storage that stores data in removable media such as SD cards, SQLite databases that store data relational databases, and the Room Persistence Library that uses SQLite databases through an abstraction layer to store data (Android Developers, n.d.). Depending on the needs of the application, one method may be more suitable than another. For example, for simple data, Shared Preferences or DataStore may be enough; on the other hand for larger datasets or complex data structures, SQLite or Room are better options. The following table describes the various Android storage options and their use cases.

Table 1

Android Data Storage Options

Storage Type	Description	Use Case	Access Method	Permissions Needed	Other Apps Access	Removed on Uninstall
Shared Preferences	Stores key-value pairs	Simple data like settings, user preferences	Jetpack Preferences library	None	No	Yes
Internal Storage	Stores files on-device	Files for app use only	getFilesDir() or getCacheDir()	None	No	Yes
External Storage (App-specific)	Stores data in removable media (SD cards)	Files for app use only	getExternalFilesDir() or getExternalCacheDir()	None (Android 4.4+)	No	Yes
External Storage (Shared)	Stores data in removable media (SD cards)	Shareable media (images, audio, videos)	MediaStore API	READ_EXTERNAL_STORAGE (Android 11+) when accessing other apps' files	Yes (with permissions)	No
SQLite Database	Stores data in relational databases	Structured data	Direct SQLite API	None	No	Yes
Room Persistence Library	Uses SQLite through an abstraction layer	Structured data with ORM benefits	Room persistence library	None	No	Yes
Documents and files	Other shareable content	Other types of shareable content	Storage Access Framework	None	Yes (via system file picker)	No

Note: The table provides descriptions of the various Android storage methods and their use cases. Data from "Data and file storage overview" by Android Developers (n.d.)

Understanding The Differences Between SQLite and Room Persistence Library

SQLite on Android can store data on the user's device (Chaitanyamunje, 2025). It is an open-source database that stores relational data in the form of tables. It is widely used, and its lightweight overhead makes it ideal for environments with limited resources such as mobile phones. SQLite uses the CRUD (Create, Read, Update, Delete) SQL approach to manipulate data.

On the other hand, Room Persistence Library is an abstraction layer built on top of SQLite, it provides an object-oriented approach to managing persistent data by automatically converting data objects (the abstraction layer) to relational data that can be stored using SQLite and converting relational to object data that can be used by the application. Room significantly reduces boilerplate code compared to straight SQLite implementations. A boilerplate is code used to perform common database operations; for example, converting between database tables and Kotlin objects.

As described above, the approaches are different, one uses a direct SQL query-based approach and the other one provides an object-oriented abstraction that handles the SQL operations. The question that can be asked is when is it better to use one approach over the other? For applications with simple data requirements probably would be due to very light overhead compared to Room. On the other hand, Room would be better for applications with complex data models or larger datasets. The table below lists the major differences between SQLite and Room Persistence Library.

Table 2

SQLite vs Room Persistence Library

Feature	SQLite	Room
Architecture	Embedded database engine	Abstraction layer over SQLite
SQL verification	Runtime	Compile-time
Object mapping	Manual	Automatic
Boilerplate code	Extensive	Minimal
Migration support	Manual	Built-in
Observability	Requires custom implementations	Supports LiveData and Flow
Learning curve	Steeper for beginners	Easier to learn and use
Flexibility	More control over SQL queries	Less direct control over SQL
Debugging	Can be more challenging	Easier to debug with compile-time checks

Note: The table lists the differences between SQLite and Room Persistence Library based on various features. From several sources (Android Developers, n.d.; Mbano, 2022; Zincircioğlu, 2023; Naniewicz, 2024)

To summarize, in mobile Applications, data persistence is the capability of an application to store data locally with the goal of being retrieved even after the program is terminated and restarted. To implement data persistence Android ecosystem offers several options such as sharing preferences, internal storage, external storage, SQLite, and the Room Persistence Library. While SQLite offers a direct, lightweight approach to managing local data, Room Persistence Library provides an object-oriented abstraction layer that reduces boilerplate code and seamlessly integrates with the Android ecosystem. Therefore, when implementing a data persistent solution is essential to understand the difference between the available storage options and how they align with the specific needs of the application.

I am not sure what the note-taking application assignment is. However, for my portfolio project, I am creating an App that makes API calls to a NoSQL database to retrieve JSON documents, which was planning to store on device. However, after learning about data persistence on mobile devices, I will probably retrieve the JSON documents with the Moshi library, and after they are translated to objects I will probably use the Room Persistence Library to store the data on device.

Alex

References:

Android Developers (n.d.). *Data and file storage overview*. Android Developer. https://developer.android.com/training/data-storage#:~:text=If%20you%20have%20data%20that's,contains%20more%20than%202%20columns).

Chaitanyamunje (2025, January 6). *How to create and add data to SQLite database in Android?* GeeksForGeeks. https://www.geeksforgeeks.org/how-to-create-and-add-data-to-sqlite-database-in-android/

Cuello, C. (2023, September 17). *What is data Persistence? A complete guide*. Rivery. https://rivery.io/data-learning-center/data-persistence/

Mbano, U. (2022, April 16). *Android Room versus SQLite — Which is best?* DVT software engineering. Medium. https://medium.com/dvt-engineering/android-room-versus-sqlite-which-is-best-32ff651bc361

MongoDB (n.d.). What is Data Persistence? MongoDb. https://www.mongodb.com/resources/basics/databases/data-persistence

Naniewicz, R., (2024, February 8). *Check out why Room is a retrofit for SQLite*. Netguru. https://www.netguru.com/blog/check-out-why-room-is-a-retrofit-for-sqlite

Zincircioğlu, S. (2023, May 25). *RoomDB vs SQLite : Exploring database options for Android development*. Medium. https://medium.com/huawei-developers/roomdb-vs-sqlite-exploring-database-options-for-android-development-1120151e6737