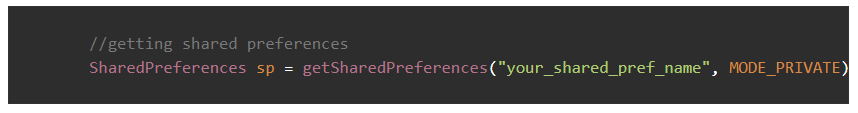
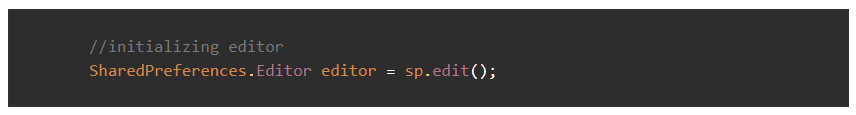
**1. Explain the steps involved in saving and retrieving data using SharedPreferences in Android.**

Ans: -

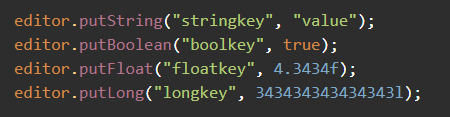
-> We have the following steps to save values in SharedPreferences.

1. **Get the SharedPreferences**:- Inside your activity, we will call the method getSharedPreferences().



2.**Initializing the Editor:-** Now in this line, where we will put the values to be saved in key-value pairs.  


**3.Put the values :-** The putDataType() method in the editor saves data using a key-value pair. The key is always a string, and the value depends on the data type. For example, putString stores a string, putInt stores an integer, and so on for other types.

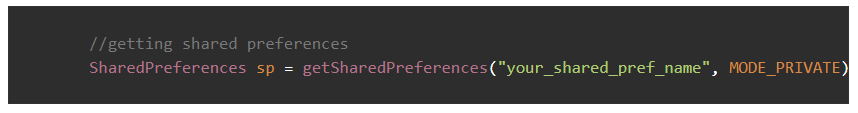


1. **Apply the changes :-** Then from the Editor instance, we will simply call the apply() method to save the changes.

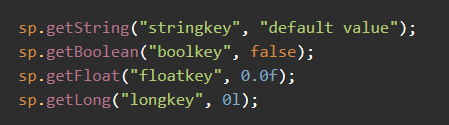


-> ****Retrieving Data:****

****1.Get SharedPreferences :** This step is the same as we did above.**



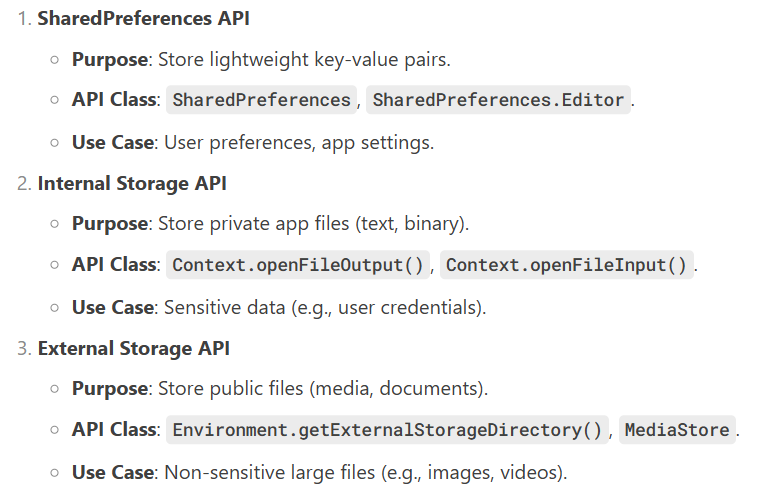
****2. Get Values:**To get the values, we have method getDataType(), i.e. getString(), getBoolean(), etc.**

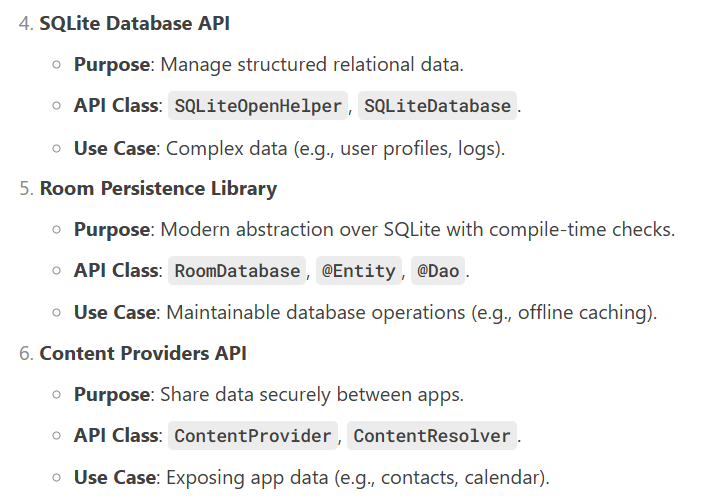


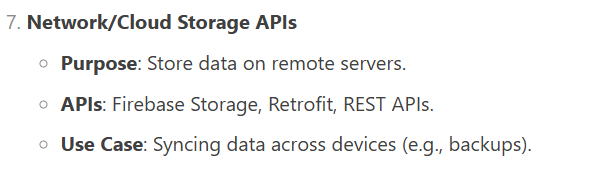
1. **Describe the different storage options available in Android and compare their use cases.**

Ans :-

**Android Storage Options & Use Cases**







**3. How can we read and write data in internal storage? Provide a code example.**

Ans:-

Android’s internal storage is private to the app and ideal for sensitive data. Below are the steps and code examples to read/write data:

* **Writing Data to Internal Storage:**

1.Open a File Stream: Use openFileOutput() with filename and mode (MODE\_PRIVATE ensures file is app-private).

2.Write Data: Convert data to bytes and write to the stream.

3.Close the Stream: Release resources.

Example:

try {

// Open file in private mode (overwrites existing file)

FileOutputStream fos = openFileOutput("my\_data.txt", Context.MODE\_PRIVATE);

String data = "Hello, Internal Storage!";

fos.write(data.getBytes()); // Write data as bytes

fos.close(); // Always close the stream

} catch (IOException e) {

e.printStackTrace();

}

* **Reading Data from Internal Storage :-**

1.Open a File Stream: Use openFileInput() with the filename.

2.Read Data: Convert bytes to readable format (e.g., String).

3.Close the Stream: Release resources.

Example:

try {

FileInputStream fis = openFileInput("my\_data.txt");

InputStreamReader isr = new InputStreamReader(fis);

BufferedReader br = new BufferedReader(isr);

StringBuilder sb = new StringBuilder();

String line;

while ((line = br.readLine()) != null) {

sb.append(line); // Read line-by-line

}

String data = sb.toString(); // Final data

br.close(); // Close streams

fis.close();

// Use 'data' (e.g., display in TextView)

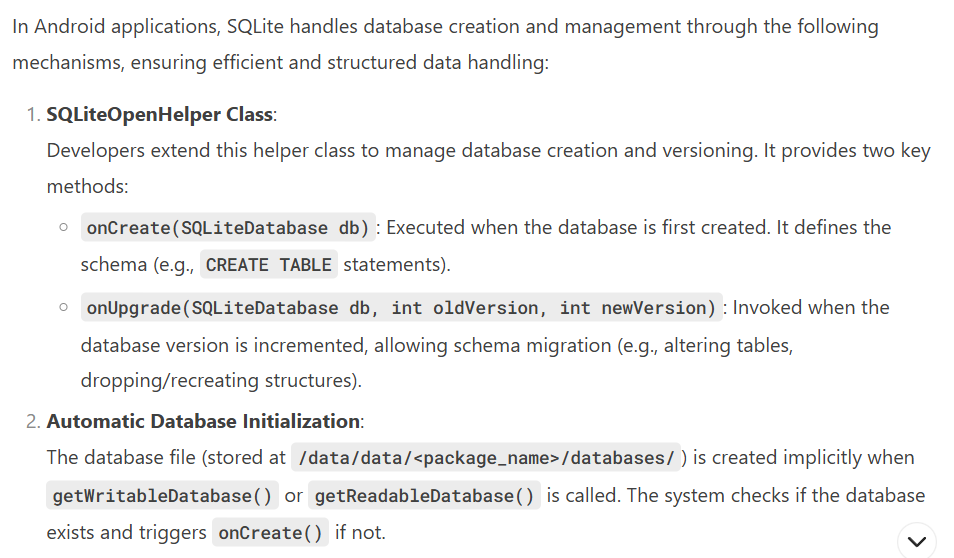
} catch (IOException e) {

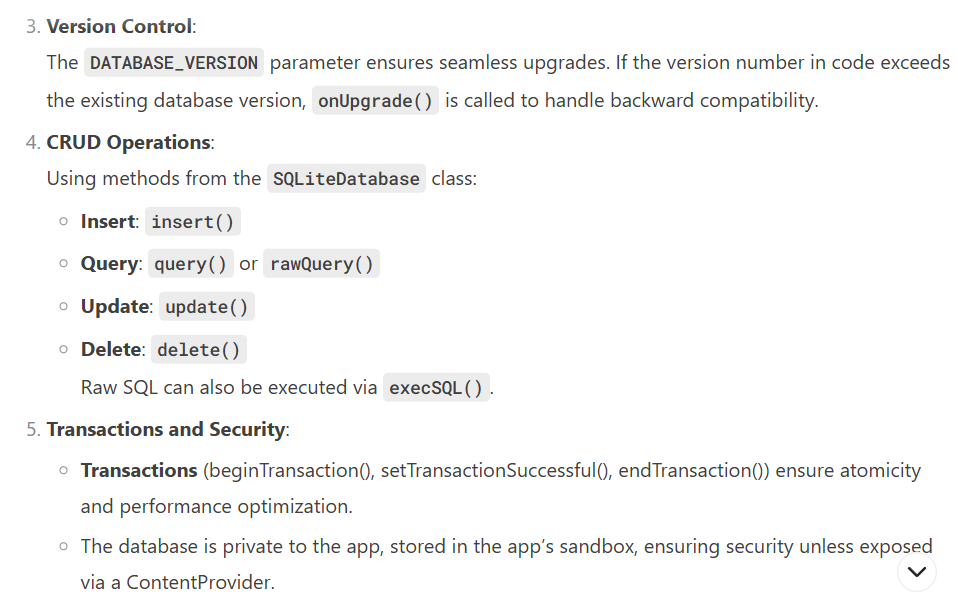
e.printStackTrace();

}

**4.How does SQLite handle database creation and management in Android applications?**

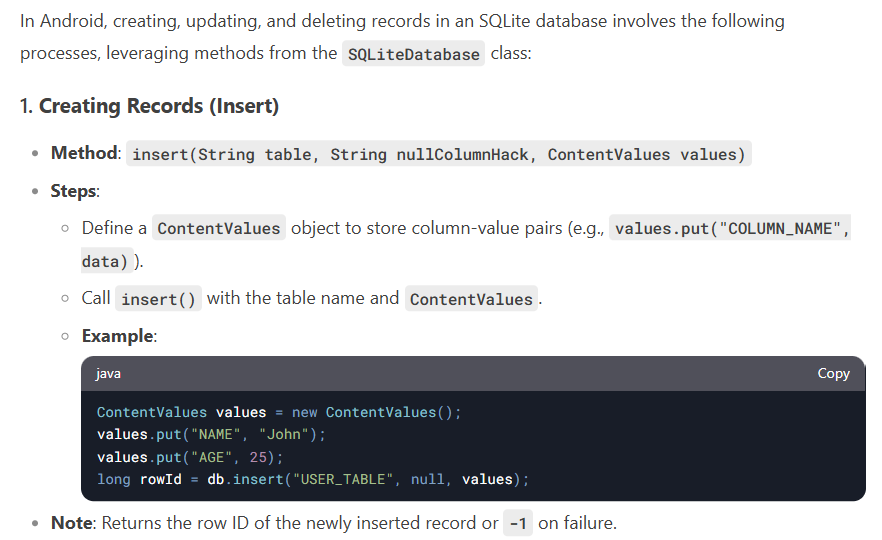
Ans :-

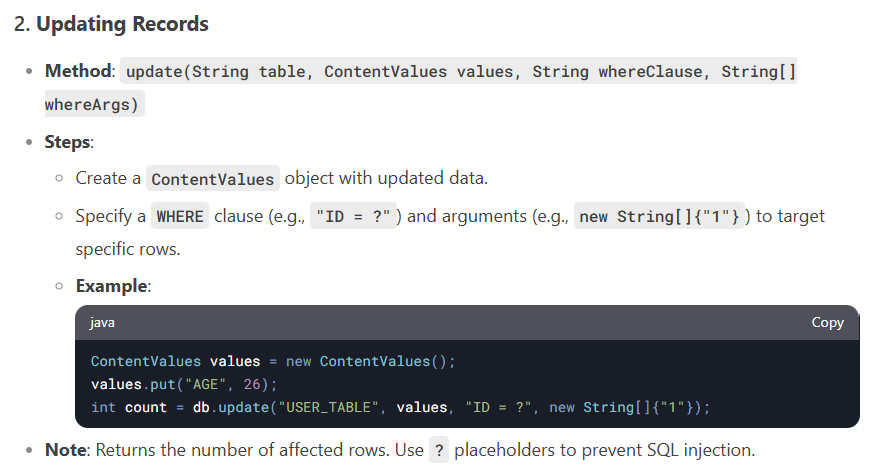


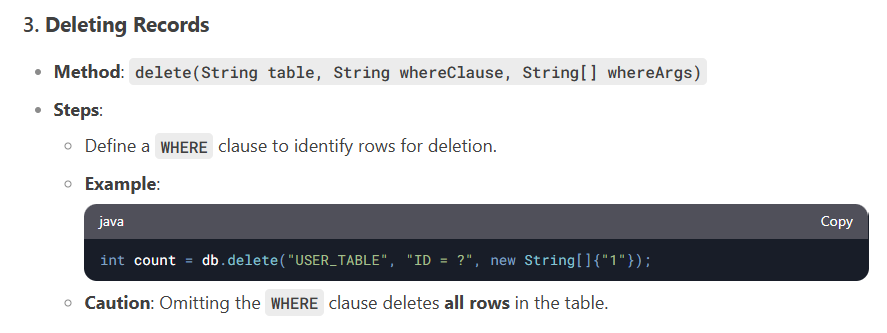


1. **Describe the process of creating, updating, and deleting records in an SQLite database.**

Ans :

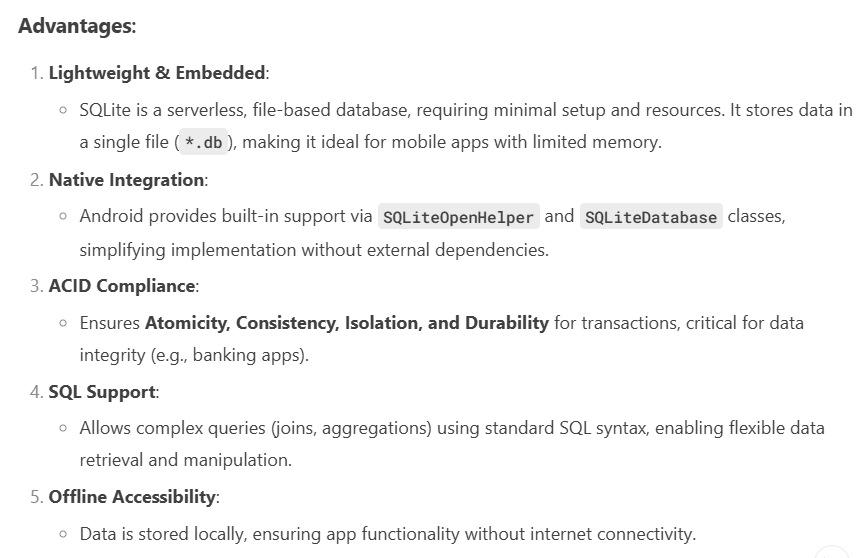


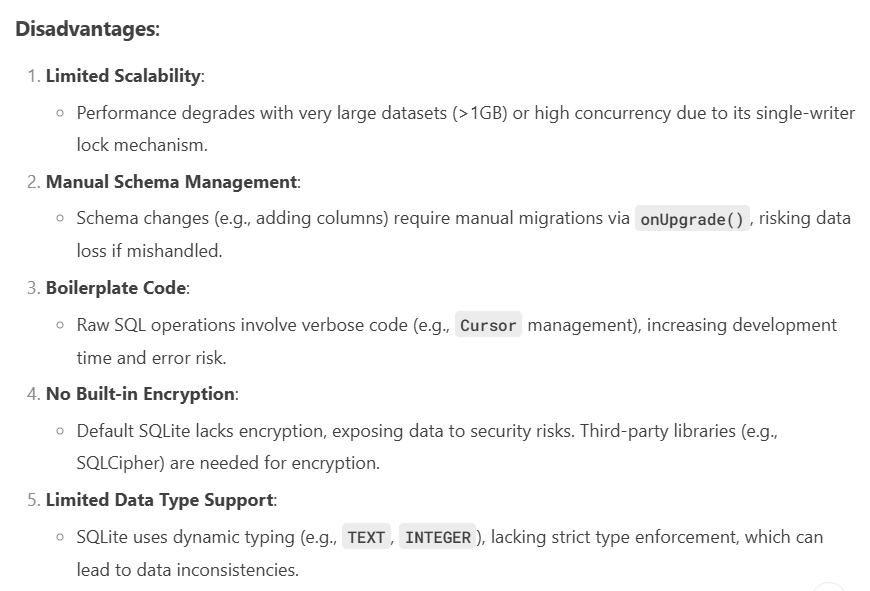




**6. Discuss the advantages and disadvantages of using SQLite for local data storage in Android.**

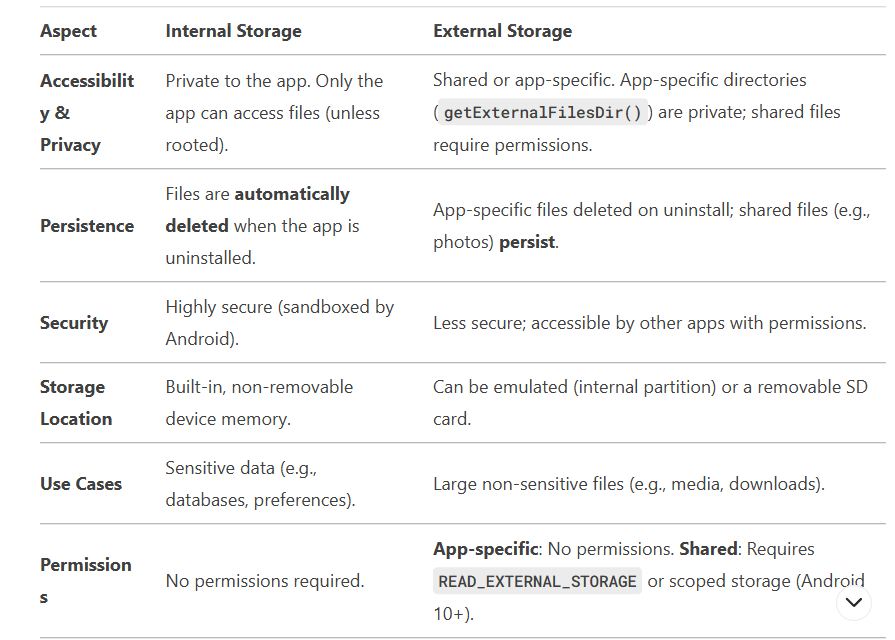
Ans :-





**7. What are the key differences between internal and external storage in Android?**

Ans :-



1. **Discuss the role of the Firebase Realtime Database in Android applications.**

Ans :-

