ISEP 2025-2026

# DATA STORAGE

## Kotlin

Valentin JENTET-GUERIN
Fabien D’ALMEIDA LERO

# Local Storage

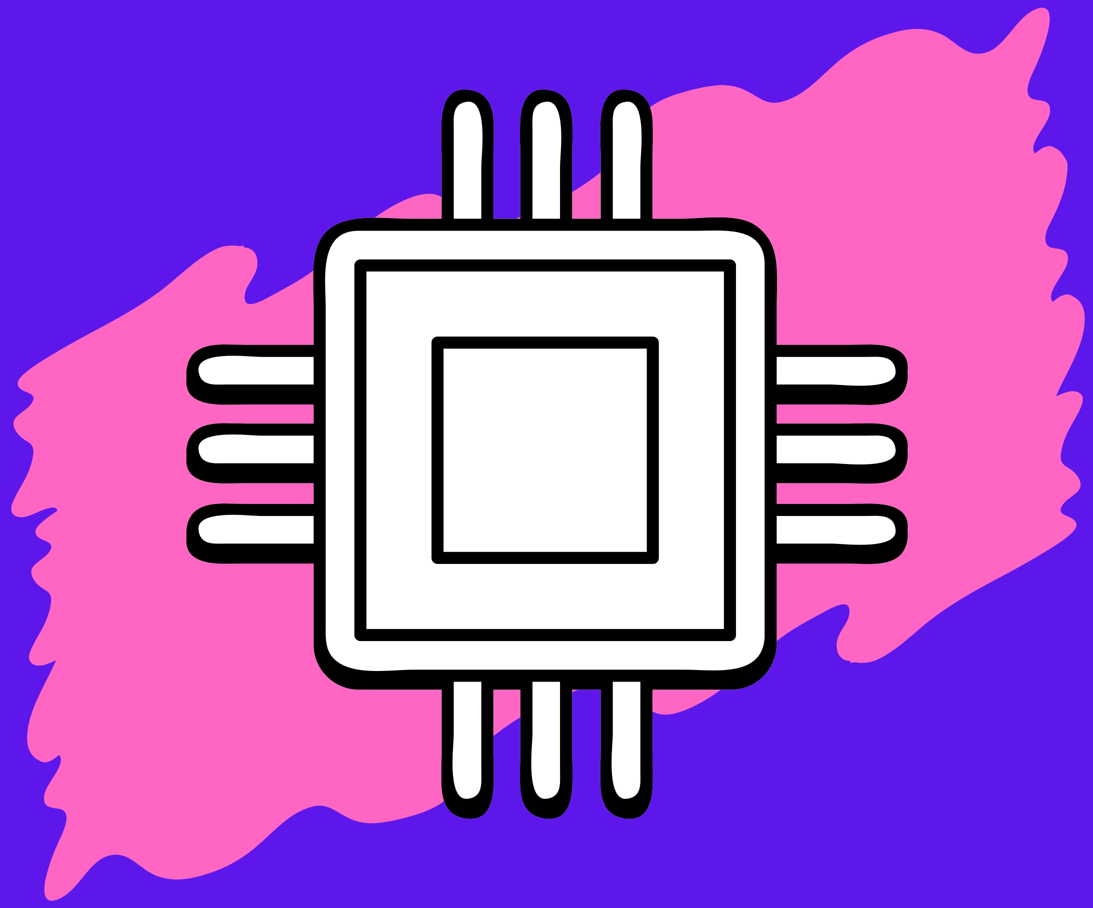
A

Local storage lets apps keep data on-device so it survives app restarts and offline use.

B

Android offers several options:

- key–value storage (for tiny settings)
- files (for raw documents/media)
- databases (for structured, queryable records).



# What is Data Versioning ?

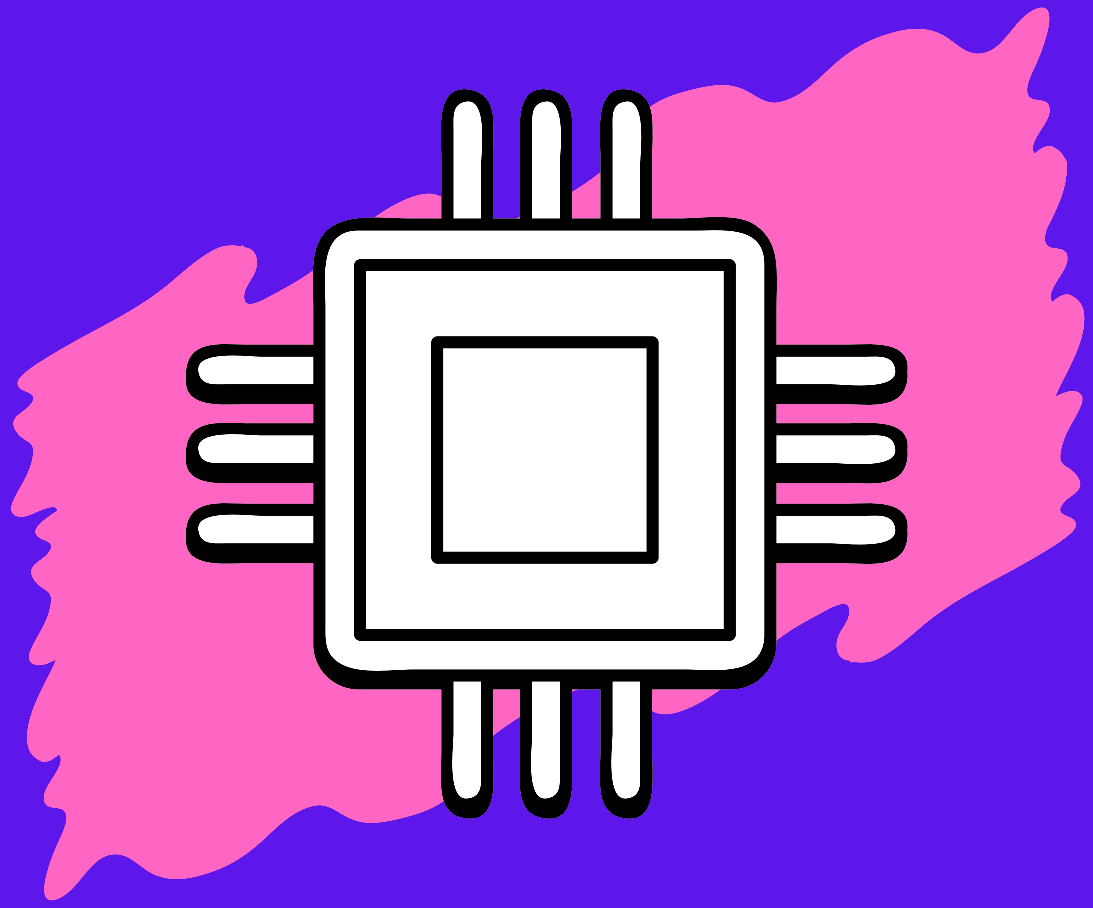
Data versioning saves every change to your data, like a history of snapshots you can access anytime.

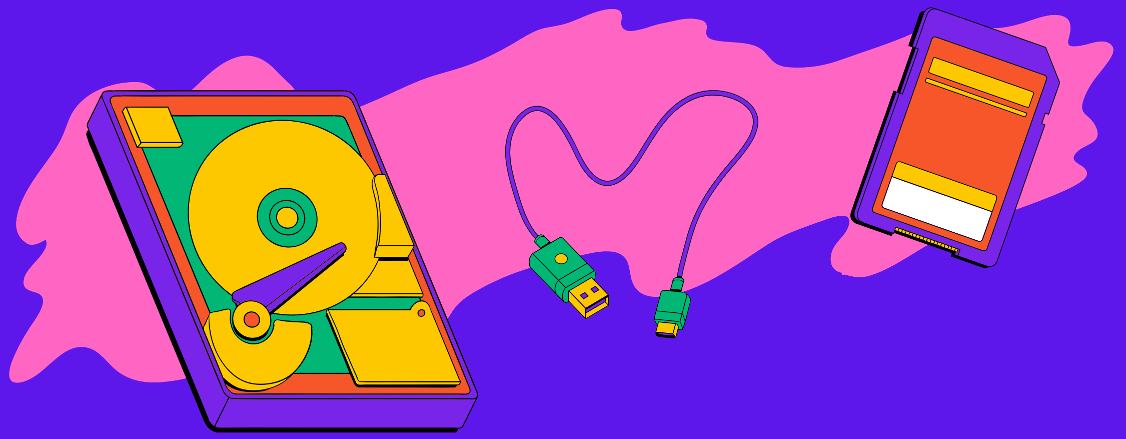
A Time-travel queries : Access historical data states

B Audit & compliance : Track who changed what and when

B Rollback capability : Revert to previous versions

B Reproducibility : Ensure consistent analysis results





# Storage Options:
# What to Use and When

A SharedPreferences

B DataStore

C SQLite

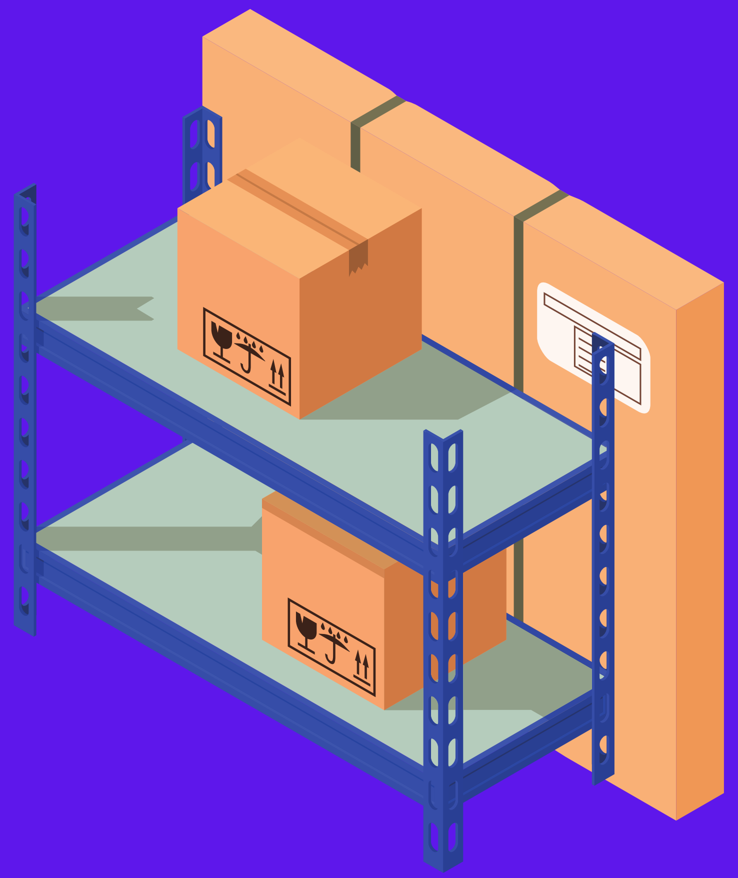
D Room

# SharedPreferences

A What it is: A simple “dictionary” saved on the device that maps a key to a value (like theme = dark).

B When to use: A few preferences or flags, not complex data models.

C Why: Zero schema work and no SQL, just save and read by key.



# DateStore

A What it is: A background-saving settings store that can stream updates as a Flow, choose simple key–value or a strongly typed model.
Example: save “language = en” and update the UI when it changes.

B When to use: Preferences and small state where you want reliability and zero UI jank.

C Why: Avoids blocking the main thread and plays well with coroutines/Flow.

# SQLite

A What it is: Like a spreadsheet stored inside the app: columns for each field, rows for each item, controlled with SQL commands.

B When to use: When data has fields and relationships and you want fast searching and sorting in the app.

C Why: Structure makes the app predictable and durable without internet.

# Room

A What it is: A helper that converts your database tables into Kotlin classes and your queries into simple interface methods, with errors flagged during build.

B When to use: If you want SQLite power but not the hassle of Cursors and manual mapping.

C Why: Less repetitive code, safer queries, and smoother updates when your database changes.

# Choose the right storage

A

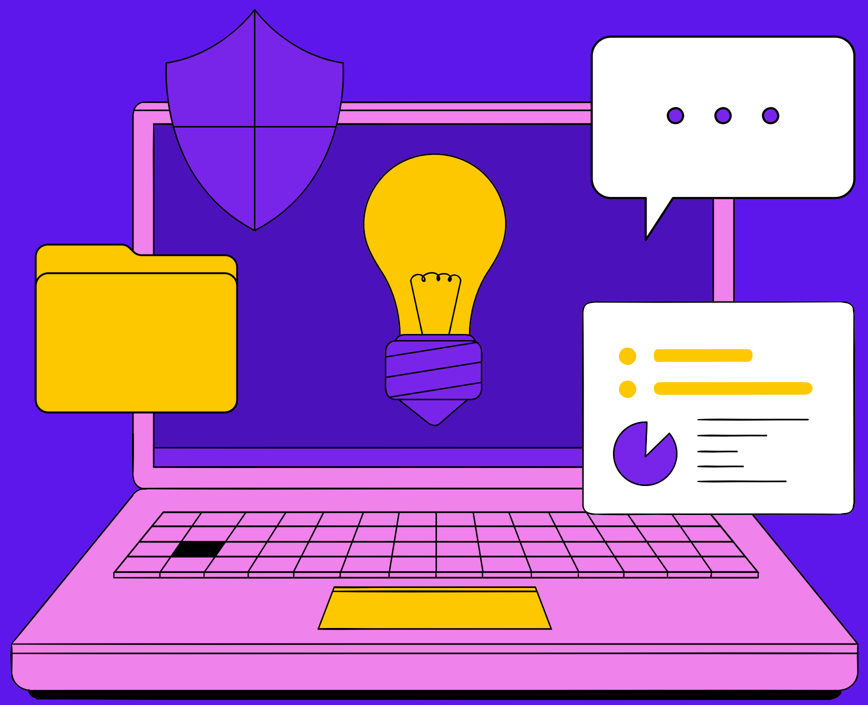
Settings/flags → DataStore

- Small key–value prefs (theme, language, toggles).

B

Lists/records → Room (or SQLite for minimal layers)

- Structured data you query, sort, filter, join.



# Demonstration

# Thank you for listening.
# Any questions?