* 使用数据库管理数据的优势
  + 检索数据时，数据库耗时较低，资源占用小（面对庞大的数据时，不可能将所有数据都载入内存，如果频繁进行IO操作，性能损耗较大）
  + 管理数据时，可以使用数据库降低一些常规操作的代码量
  + 数据库普遍支持增量写入，安全性较好
  + 主流数据库性能普遍较好
* 数据搜索服务：在单个设备上，为应用程序提供搜索引擎级的全文索引管理、建立索引和搜索功能
* 数据存储管理：为应用开发者提供系统存储路径、存储设备列表，存储设备属性的查询和管理功能

|  |  |  |  |
| --- | --- | --- | --- |
| **本地数据库对比** | | | |
| 数据库对比 | 关系型数据库 | 对象关系映射数据库 | 轻量级偏好数据库 |
| 特点 | * 基于SQLite * 使用SQL语句操作数据库 | * 基于SQLite * 使用对象接口操作数据库 | * NoSQL数据库，以键值对形式保存数据 * 在内存中维护一份实例 |
| 优点 | 性能好 | 自动将实体对象与数据表进行映射，提高了开发效率 | * 速度快 * 易于开发 |
| 缺点 | * 开发步骤繁琐易出错 * 频繁操作会导致性能下降 | 无法直接通过SQL语句操作数据库导致性能降低 | 数据复杂或数据量较大时会极大降低性能 |

* 关系型数据库（使用SQLite作为持久化存储引擎）
  + 在HarmonyOS中，只有关系型数据库可以使用SQL语句操作数据库
  + 基于 SQLite 组件提供了一套完整的对本地数据库进行管理的机制
  + 对外提供增、删、改、查接口，也可以直接运行输入的SQL语句
  + 运作机制
    - 对外提供通用的操作接口
    - 底层使用 SQLite 作为持久化存储引擎
    - 支持 SQLite 具有的所有数据库特性
* 对象关系映射数据库（使用SQLite作为持久化存储引擎）
  + 对象关系映射数据库是一款基于SQLite 的数据库框架，屏蔽了底层 SQLite 数据库的 SQL 操作，针对实体和关系提供了增删改查等一系列的面向对象接口
  + 应用开发者不必再去编写复杂的 SQL 语句， 以操作对象的形式来操作数据库，提升效率的同时也能聚焦于业务开发
  + 对象关系映射数据库主要包括三个组件
    - 数据库：被开发者用 @Database 注解，且继承了 OrmDatabase 的类，对应关系型数据库
    - 实体对象：被开发者用 @Entity 注解，且继承了 OrmObject 的类，对应关系型数据库中的表
    - 对象数据操作接口：包括数据库操作的入口 OrmContext 类和谓词接口(OrmPredicate) 等
  + 运作机制
    - 对象关系映射数据库操作基于关系型数据库操作接口实现，并在其基础上实现了对象关系映射等特性（操作数据库）
    - 使用前需要先配置实体模型与关系映射文件，类生成工具会解析这些文件，生成数据库帮助类（生成实体对象）
    - 框架在运行时，会根据配置创建数据库，并在存储过程中自动完成对象关系映射（生成数据库）
* 轻量级偏好数据库（一种支持轻量级Key-Value操作的非关系型数据库）
  + 轻量级偏好数据库主要提供Key-Value操作，支持本地应用存储少量数据，数据存储在本地文件中，同时也加载在内存中，所以访问速度更快，效率更高
    - Key-Value数据库：一种以键值对存储数据的数据库，类似Java中的map，Key是关键字，Value是值
    - 是非关系型数据库：不采用关系模型来组织数据，数据之间无关系，扩展性好
  + 轻量级偏好数据库是轻量级存储，主要用于保存应用的一些常用配置，并不适合存储大量数据和频繁改变数据的场景
  + 运作机制
    - 应用通过偏好型数据库操作类完成数据库操作
    - 应用借助DatabaseHelper API将指定文件的内容加载到Preferences实例，该实例会存储在内存中，直到应用主动从内存中移除该实例或者删除该文件
    - 应用使用Preferences API进行读写时，通过flush或者flushSync将Preferences实例持久化