SQLite [Kotlin]

Trabajar con Kotlin y SQLite no es tan diferente como hacerlo con Java. Se necesita crear una serie de clases que gestionarán las tablas, y su lógica es similar. Inicialmente, crearemos una clase que **DBHelper**. En ella definiremos en companion object con los datos necesarios y sus métodos onCreate y onUpgrade.

```
class DBHelper (contexto : Context, name : String, factory : SQLiteDatabase.CursorFactory?,
                            version : Int) : SQLiteOpenHelper (contexto, name, factory, version){

    companion object{
        val DB_NAME = "miBaseDeDatos.db"
        val DB_VERSION = 1
        val TABLE_NAME = "t_ciudades"

        val ID_COL = "id"
        val ID_TEXT = "text"
    }

    override fun onCreate(db: SQLiteDatabase?) {
        val CREATE_DB = ("CREATE TABLE " + TABLE_NAME + " ("
                + ID_COL + " INTEGER PRIMARY KEY AUTOINCREMENT, "
                + ID_TEXT + " TEXT" + ")")
        db?.execSQL(CREATE_DB)
    }

    override fun onUpgrade(db: SQLiteDatabase?, oldVersion: Int, newVersion: Int) {
        db?.execSQL( sql: "DROP TABLE IF EXISTS " + TABLE_NAME)
        onCreate(db)
    }
}
```

Igual que antes, en cuanto se instancie esta clase se determinará si la base de datos existe o no. Si no lo está, se ejecutará el método **onCreate ()**. Si ya existe, se comprobará la versión de la base de datos. Si es una versión anterior, se ejecutará **onUpgrade ()**, donde podremos indicar las modificaciones oportunas. Si las versiones coinciden y la base de datos existe, simplemente se devuelve una conexión.

Puede suceder que nuestra app cuando sea actualizada termine con cambios en la base de datos. En estos casos, es posible gestionar y actualizar las **versiones** de la base de datos de la siguiente forma:

```
override fun onUpgrade(db: SQLiteDatabase?, oldVersion: Int, newVersion: Int)

    if (oldVersion == 1)
        db.execSQL ("ALTER TABLE t_ciudades ADD COLUMN poblacion INTEGER")

    if (oldVersion <= 1)
        ...

    if (oldVersion > 4)
        throw IllegalStateException ("Version desconocida $oldVersion")
}
```

Por último, solamente faltaría implementar los métodos de acceso que deseemos utilizar contra nuestra base de datos. Por ejemplo:

```
// Ej: un select *
fun getName(): Cursor? {
    val db = this.readableDatabase
    return db.rawQuery( sql: "SELECT * FROM " + TABLE_NAME,  selectionArgs: null)
}
```

Práctica 38

[Kotlin] Realiza una app que sea capaz de almacenar sus canciones favoritas. Se guardará el título de la canción, el autor, y la URL de YouTube donde se puede escuchar. La app debe permitir listar todas las canciones, añadir una canción, borrar una canción y modificar sus datos. Las canciones deberán aparecer en formato lista.

Práctica 39

[Kotlin] Lo mismo que antes, pero al pulsar una canción de la lista se hará un *intent* y se reproducirá la canción en el navegador.

Práctica 40

[Kotlin] Lo mismo que antes, pero al pulsar una canción de la lista se hará un *intent* y se reproducirá la canción en otro activity que tendrá un componente **Media Player**. Este ejercicio propuesto es para que investigues un rato si ya has terminado otras tareas. Esto lo veremos más adelante. Que te diviertas.