參考課本題目

<?xml version="1.0" encoding="utf-8"?>

<androidx.constraintlayout.widget.ConstraintLayout

xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".MainActivity">

<TextView

android:id="@+id/textView"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginStart="16dp"

android:text="書名 :"

android:textColor="@android:color/black"

android:textSize="22sp"

app:layout\_constraintBottom\_toBottomOf="@+id/ed\_book"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toTopOf="@+id/ed\_book" />

<EditText

android:id="@+id/ed\_book"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_marginStart="8dp"

android:layout\_marginTop="16dp"

android:layout\_marginEnd="8dp"

android:ems="10"

android:inputType="textPersonName"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toEndOf="@+id/textView"

app:layout\_constraintTop\_toTopOf="parent" />

<TextView

android:id="@+id/textView2"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="價格 :"

android:textColor="@android:color/black"

android:textSize="22sp"

app:layout\_constraintBottom\_toBottomOf="@+id/ed\_price"

app:layout\_constraintStart\_toStartOf="@+id/textView"

app:layout\_constraintTop\_toTopOf="@+id/ed\_price" />

<EditText

android:id="@+id/ed\_price"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_marginTop="16dp"

android:ems="10"

android:inputType="number"

app:layout\_constraintEnd\_toEndOf="@+id/ed\_book"

app:layout\_constraintStart\_toStartOf="@+id/ed\_book"

app:layout\_constraintTop\_toBottomOf="@+id/ed\_book" />

<LinearLayout

android:id="@+id/linearLayout"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_marginStart="8dp"

android:layout\_marginTop="8dp"

android:layout\_marginEnd="8dp"

android:orientation="horizontal"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toBottomOf="@+id/ed\_price">

<Button

android:id="@+id/btn\_insert"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:layout\_marginEnd="4dp"

android:text="新增" />

<Button

android:id="@+id/btn\_update"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:layout\_marginEnd="4dp"

android:text="修改" />

<Button

android:id="@+id/btn\_delete"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:layout\_marginEnd="4dp"

android:text="刪除" />

<Button

android:id="@+id/btn\_query"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="查詢" />

</LinearLayout>

<ListView

android:id="@+id/listView"

android:layout\_width="0dp"

android:layout\_height="0dp"

android:layout\_marginStart="8dp"

android:layout\_marginTop="8dp"

android:layout\_marginEnd="8dp"

android:layout\_marginBottom="8dp"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintBottom\_toBottomOf="parent"

app:layout\_constraintTop\_toBottomOf="@+id/linearLayout" />

</androidx.constraintlayout.widget.ConstraintLayout>

**選寫MyDBHelper， 需要建立myTable資料夾，包含book字串欄位、一個price整數欄位。**

import android.content.Context

import android.database.sqlite.SQLiteDatabase

import android.database.sqlite.SQLiteOpenHelper

//自訂建構子並繼承 SQLiteOpenHelper 類別

class MyDBHelper (

context: Context,

name: String = database,

factory: SQLiteDatabase.CursorFactory? = null,

version: Int = v

) : SQLiteOpenHelper(context, name, factory, version) {

companion object {

private const val database = "myDatabase" //資料庫名稱

private const val v = 1 //資料庫版本

}

override fun onCreate(db: SQLiteDatabase) {

//建立 myTable 資料表，表內有 book 字串欄位和 price 整數欄位

db.execSQL("CREATE TABLE myTable(book text PRIMARY KEY, price integer NOT NULL)")

}

override fun onUpgrade(db: SQLiteDatabase, oldVersion: Int,

newVersion: Int) {

//升級資料庫版本時，刪除舊資料表，並重新執行 onCreate()，建立新資料表

db.execSQL("DROP TABLE IF EXISTS myTable")

onCreate(db)

}

}

**選寫**MainActivity**， 需要建立MyDBHelper實體，並透過writableDatabase來取得SQLiteDatabase實體。**

import android.database.sqlite.SQLiteDatabase

import androidx.appcompat.app.AppCompatActivity

import android.os.Bundle

import android.widget.\*

class MainActivity : AppCompatActivity() {

private var items: ArrayList<String> = ArrayList()

private lateinit var adapter: ArrayAdapter<String>

private lateinit var dbrw: SQLiteDatabase

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity\_main)

//取得資料庫實體

dbrw = MyDBHelper(this).writableDatabase

//宣告 Adapter 並連結 ListView

adapter = ArrayAdapter(this,

android.R.layout.simple\_list\_item\_1, items)

findViewById<ListView>(R.id.listView).adapter = adapter

//設定監聽器

setListener()

}

override fun onDestroy() {

dbrw.close() //關閉資料庫

super.onDestroy()

}

**為4個按鈕分別建立監聽事件，並在按下時執行對應的副程式。**

//設定監聽器

private fun setListener() {

val ed\_book = findViewById<EditText>(R.id.ed\_book)

val ed\_price = findViewById<EditText>(R.id.ed\_price)

findViewById<Button>(R.id.btn\_insert).setOnClickListener {

//判斷是否有填入書名或價格

if (ed\_book.length() < 1 || ed\_price.length() < 1)

showToast("欄位請勿留空")

else

try {

//新增一筆書籍紀錄於 myTable 資料表

dbrw.execSQL(

"INSERT INTO myTable(book, price) VALUES(?,?)",

arrayOf(ed\_book.text.toString(),

ed\_price.text.toString())

)

showToast("新增:${ed\_book.text},價格:${ed\_price.text}")

cleanEditText()

} catch (e: Exception) {

showToast("新增失敗:$e")

}

}

findViewById<Button>(R.id.btn\_update).setOnClickListener {

//判斷是否有填入書名或價格

if (ed\_book.length() < 1 || ed\_price.length() < 1)

showToast("欄位請勿留空")

else

try {

//尋找相同書名的紀錄並更新 price 欄位的值

dbrw.execSQL("UPDATE myTable SET price = ${ed\_price.text} WHERE book LIKE '${ed\_book.text}'")

showToast("更新:${ed\_book.text},價格:${ed\_price.text}")

cleanEditText()

} catch (e: Exception) {

showToast("更新失敗:$e")

}

}

findViewById<Button>(R.id.btn\_delete).setOnClickListener {

//判斷是否有填入書名

if (ed\_book.length() < 1)

showToast("書名請勿留空")

else

try {

//從 myTable 資料表刪除相同書名的紀錄

dbrw.execSQL("DELETE FROM myTable WHERE book LIKE '${ed\_book.text}'")

showToast("刪除:${ed\_book.text}")

cleanEditText()

} catch (e: Exception) {

showToast("刪除失敗:$e")

}

}

findViewById<Button>(R.id.btn\_query).setOnClickListener {

//若無輸入書名則 SQL 語法為查詢全部書籍，反之查詢該書名資料

val queryString = if (ed\_book.length() < 1)

"SELECT \* FROM myTable"

else

"SELECT \* FROM myTable WHERE book LIKE '${ed\_book.text}'"

val c = dbrw.rawQuery(queryString, null)

c.moveToFirst() //從第一筆開始輸出

items.clear() //清空舊資料

showToast("共有${c.count}筆資料")

for (i in 0 until c.count) {

//加入新資料

items.add("書名:${c.getString(0)}\t\t\t\t 價格:${c.getInt(1)}")

c.moveToNext() //移動到下一筆

}

adapter.notifyDataSetChanged() //更新列表資料

c.close() //關閉 Cursor

}

}

//建立 showToast 方法顯示 Toast 訊息

private fun showToast(text: String) =

Toast.makeText(this,text, Toast.LENGTH\_LONG).show()

//清空輸入的書名與價格

private fun cleanEditText() {

findViewById<EditText>(R.id.ed\_book).setText("")

findViewById<EditText>(R.id.ed\_price).setText("")

}

}