Сначала:

 Нарисуй простую иконку в Paint (например, значок базы данных или действия).

 Сохрани её как PNG-файл, например, ic\_database.png.

 Помести файл в папку app/src/main/res/drawable.

#### Оценка 3:

* Реализовано создание таблицы при первом запуске (DatabaseHelper.onCreate).
* Кнопка INSERT добавляет пару ключ-значение в таблицу.
* Кнопка SELECT выполняет выборку значения по ключу и отображает его в поле Value.
* Интерфейс соответствует скриншоту, компоненты имеют осмысленные ID: txt\_key, txt\_value, button\_insert, и т.д.

#### Оценка 4:

* Текстовые надписи вынесены в strings.xml.
* Реализовано удаление записи через кнопку DELETE.
* При удалении отображается диалоговое окно AlertDialog с подтверждением.

#### Оценка 5:

* Реализовано обновление записи через кнопку UPDATE.
* Добавлена обработка ошибок:
  + При добавлении существующего ключа: "Key already exists!".
  + При обновлении/удалении несуществующего ключа: "Key not found!".
  + При пустом ключе: "Key cannot be empty!".
* Для всех диалоговых окон установлен заголовок и иконка (ic\_database).
* (ic\_delete).

Переходим к кодам, их просто вставить и будет счастье.

Activity\_main.xaml:

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 android:padding="16dp">  
  
 <!-- Поле Key -->  
 <TextView  
 android:id="@+id/txt\_key\_label"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/lab\_key" />  
 <EditText  
 android:id="@+id/txt\_key"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="@string/lab\_key" />  
  
 <!-- Поле Value -->  
 <TextView  
 android:id="@+id/txt\_value\_label"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/lab\_value" />  
 <EditText  
 android:id="@+id/txt\_value"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="@string/lab\_value" />  
  
 <!-- Кнопки -->  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal">  
 <Button  
 android:id="@+id/button\_insert"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:text="@string/lab\_insert" />  
 <Button  
 android:id="@+id/button\_update"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:text="@string/lab\_update" />  
 <Button  
 android:id="@+id/button\_select"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:text="@string/lab\_select" />  
 <Button  
 android:id="@+id/button\_delete"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:text="@string/lab\_delete" />  
 </LinearLayout>  
  
</LinearLayout>

String.xaml:

<resources>  
 <string name="app\_name">Database</string>  
 <string name="lab\_key">Key</string>  
 <string name="lab\_value">Value</string>  
 <string name="lab\_insert">INSERT</string>  
 <string name="lab\_update">UPDATE</string>  
 <string name="lab\_select">SELECT</string>  
 <string name="lab\_delete">DELETE</string>  
 <string name="dialog\_confirm\_delete\_title">Confirm Deletion</string>  
 <string name="dialog\_confirm\_delete\_message">Are you sure you want to delete this key?</string>  
 <string name="dialog\_confirm\_yes">Yes</string>  
 <string name="dialog\_confirm\_no">No</string>  
 <string name="error\_key\_exists">Key already exists!</string>  
 <string name="error\_key\_not\_found">Key not found!</string>  
 <string name="error\_empty\_key">Key cannot be empty!</string>  
</resources>

Cоздать DatabaseHelper.java:

package com.example.database;  
  
import android.content.ContentValues;  
import android.content.Context;  
import android.database.Cursor;  
import android.database.sqlite.SQLiteDatabase;  
import android.database.sqlite.SQLiteOpenHelper;  
  
public class DatabaseHelper extends SQLiteOpenHelper {  
  
 private static final String *DATABASE\_NAME* = "KeyValueDB";  
 private static final int *DATABASE\_VERSION* = 1;  
 private static final String *TABLE\_NAME* = "key\_value\_table";  
 private static final String *COLUMN\_KEY* = "key\_column";  
 private static final String *COLUMN\_VALUE* = "value\_column";  
  
 public DatabaseHelper(Context context) {  
 super(context, *DATABASE\_NAME*, null, *DATABASE\_VERSION*);  
 }  
  
 @Override  
 public void onCreate(SQLiteDatabase db) {  
 // Создание таблицы при первом запуске  
 String createTable = "CREATE TABLE " + *TABLE\_NAME* + " (" +  
 *COLUMN\_KEY* + " TEXT PRIMARY KEY, " +  
 *COLUMN\_VALUE* + " TEXT)";  
 db.execSQL(createTable);  
 }  
  
 @Override  
 public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {  
 db.execSQL("DROP TABLE IF EXISTS " + *TABLE\_NAME*);  
 onCreate(db);  
 }  
  
 // Вставка новой записи  
 public boolean insertData(String key, String value) {  
 SQLiteDatabase db = this.getWritableDatabase();  
 ContentValues values = new ContentValues();  
 values.put(*COLUMN\_KEY*, key);  
 values.put(*COLUMN\_VALUE*, value);  
 long result = db.insert(*TABLE\_NAME*, null, values);  
 return result != -1; // Возвращает true, если вставка успешна  
 }  
  
 // Проверка, существует ли ключ  
 public boolean keyExists(String key) {  
 SQLiteDatabase db = this.getReadableDatabase();  
 Cursor cursor = db.rawQuery("SELECT \* FROM " + *TABLE\_NAME* + " WHERE " + *COLUMN\_KEY* + "=?", new String[]{key});  
 boolean exists = cursor.getCount() > 0;  
 cursor.close();  
 return exists;  
 }  
  
 // Выборка значения по ключу  
 public String selectData(String key) {  
 SQLiteDatabase db = this.getReadableDatabase();  
 Cursor cursor = db.rawQuery("SELECT " + *COLUMN\_VALUE* + " FROM " + *TABLE\_NAME* + " WHERE " + *COLUMN\_KEY* + "=?", new String[]{key});  
 if (cursor.moveToFirst()) {  
 String value = cursor.getString(0);  
 cursor.close();  
 return value;  
 }  
 cursor.close();  
 return null;  
 }  
  
 // Обновление значения по ключу  
 public boolean updateData(String key, String value) {  
 SQLiteDatabase db = this.getWritableDatabase();  
 ContentValues values = new ContentValues();  
 values.put(*COLUMN\_VALUE*, value);  
 int rowsAffected = db.update(*TABLE\_NAME*, values, *COLUMN\_KEY* + "=?", new String[]{key});  
 return rowsAffected > 0;  
 }  
  
 // Удаление записи по ключу  
 public boolean deleteData(String key) {  
 SQLiteDatabase db = this.getWritableDatabase();  
 int rowsAffected = db.delete(*TABLE\_NAME*, *COLUMN\_KEY* + "=?", new String[]{key});  
 return rowsAffected > 0;  
 }  
}

MainActivity.java

package com.example.database;  
  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;  
import android.widget.EditText;  
import androidx.appcompat.app.AlertDialog;  
import androidx.appcompat.app.AppCompatActivity;  
  
public class MainActivity extends AppCompatActivity {  
  
 private EditText txtKey, txtValue;  
 private Button buttonInsert, buttonUpdate, buttonSelect, buttonDelete;  
 private DatabaseHelper dbHelper;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 // Инициализация компонентов  
 txtKey = findViewById(R.id.*txt\_key*);  
 txtValue = findViewById(R.id.*txt\_value*);  
 buttonInsert = findViewById(R.id.*button\_insert*);  
 buttonUpdate = findViewById(R.id.*button\_update*);  
 buttonSelect = findViewById(R.id.*button\_select*);  
 buttonDelete = findViewById(R.id.*button\_delete*);  
  
 // Инициализация базы данных  
 dbHelper = new DatabaseHelper(this);  
  
 // Обработчик кнопки INSERT  
 buttonInsert.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 String key = txtKey.getText().toString().trim();  
 String value = txtValue.getText().toString().trim();  
  
 if (key.isEmpty()) {  
 showErrorDialog(getString(R.string.*error\_empty\_key*));  
 return;  
 }  
  
 if (dbHelper.keyExists(key)) {  
 showErrorDialog(getString(R.string.*error\_key\_exists*));  
 return;  
 }  
  
 if (dbHelper.insertData(key, value)) {  
 txtKey.setText("");  
 txtValue.setText("");  
 }  
 }  
 });  
  
 // Обработчик кнопки SELECT  
 buttonSelect.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 String key = txtKey.getText().toString().trim();  
 if (key.isEmpty()) {  
 showErrorDialog(getString(R.string.*error\_empty\_key*));  
 return;  
 }  
  
 String value = dbHelper.selectData(key);  
 if (value != null) {  
 txtValue.setText(value);  
 } else {  
 showErrorDialog(getString(R.string.*error\_key\_not\_found*));  
 }  
 }  
 });  
  
 // Обработчик кнопки UPDATE  
 buttonUpdate.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 String key = txtKey.getText().toString().trim();  
 String value = txtValue.getText().toString().trim();  
  
 if (key.isEmpty()) {  
 showErrorDialog(getString(R.string.*error\_empty\_key*));  
 return;  
 }  
  
 if (!dbHelper.keyExists(key)) {  
 showErrorDialog(getString(R.string.*error\_key\_not\_found*));  
 return;  
 }  
  
 if (dbHelper.updateData(key, value)) {  
 txtKey.setText("");  
 txtValue.setText("");  
 }  
 }  
 });  
  
 // Обработчик кнопки DELETE  
 buttonDelete.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 String key = txtKey.getText().toString().trim();  
 if (key.isEmpty()) {  
 showErrorDialog(getString(R.string.*error\_empty\_key*));  
 return;  
 }  
  
 if (!dbHelper.keyExists(key)) {  
 showErrorDialog(getString(R.string.*error\_key\_not\_found*));  
 return;  
 }  
  
 showDeleteDialog(key);  
 }  
 });  
 }  
  
 // Диалог ошибки  
 private void showErrorDialog(String message) {  
 new AlertDialog.Builder(this)  
 .setTitle("Error")  
 .setIcon(R.drawable.*ic\_database*)  
 .setMessage(message)  
 .setPositiveButton("OK", null)  
 .show();  
 }  
  
 // Диалог подтверждения удаления  
 private void showDeleteDialog(final String key) {  
 new AlertDialog.Builder(this)  
 .setTitle(getString(R.string.*dialog\_confirm\_delete\_title*))  
 .setIcon(R.drawable.*ic\_database*)  
 .setMessage(getString(R.string.*dialog\_confirm\_delete\_message*))  
 .setPositiveButton(getString(R.string.*dialog\_confirm\_yes*), (dialog, which) -> {  
 if (dbHelper.deleteData(key)) {  
 txtKey.setText("");  
 txtValue.setText("");  
 }  
 })  
 .setNegativeButton(getString(R.string.*dialog\_confirm\_no*), null)  
 .show();  
 }  
}