Persistència: BASE DE DADES

Gestió d'una base de dades amb Android.

Crearem una aplicació per gestionar una base de dades, la base de dades pot esser de qualsevol cosa del vostre interès, en l'exemple la base de dades gestiona una petita aplicació per guardar informació i valoració personal de vins.

Amb Android hi ha dues classes que ens ajuden amb la gestió d'una base de dades SQLite:

SQLiteOpenHelper: Per crear, obrir i controlar els canvis que feim a la base de dades durant el desenvolupament.

SQLiteDatabase: Implementa els mètodes per executar comandes SQL.

Com a bona pràctica ens convé primer crear classes per modelar les nostres taules de la base de dades, en el nostre cas crearem una classe vi:

```
public class Vi {
   private long id;
   private String nomVi;
   private String anada;
   private String tipus;
   private String lloc;
   private String graduacio;
   private String data;
   private String comentari;
   private long idBodega;
   private long idDenominacio;
   private double preu;
   private String valOlfativa;
   private String valGustativa;
   private String valVisual;
   private int nota;
   private String foto;

   public Vi() {    this.id=-1;   }

   public long getId() {  return id;   }
   public void setId(long id) {  this.id = id;   }
   public String getNomVi() {  return nomVi;   }
   public void setNomVi(String nomVi) {  this.nomVi = nomVi;   }
   public String getAnada() {  return anada;   }
   public void setAnada(String anada) {  this.anada = anada;   }
   public String getLloc() {  return lloc;   }
   public void setLloc(String lloc) {  this.lloc = lloc;   }
   public String getGraduacio() {  return graduacio;   }
   public void setGraduacio(String graduacio) {   this.graduacio = graduacio;   }
   public String getData() {  return data;   }
   public void setData(String data) {  this.data = data;   }
   public String getComentari() {  return comentari;   }
   public void setComentari(String comentari) {   this.comentari = comentari;   }
   public long getIdBodega() {  return idBodega;   }
   public void setIdBodega(long idBodega) {  this.idBodega = idBodega;   }
   public long getIdDenominacio() {  return idDenominacio;   }
   public void setIdDenominacio(long idDenominacio) {  this.idDenominacio = idDenominacio;   }
   public double getPreu() {  return preu;   }
   public void setPreu(double preu) { this.preu = preu;   }
   public String getValOlfativa() { return valOlfativa;   }
   public void setValOlfativa(String valOlfativa) { this.valOlfativa = valOlfativa;   }
   public String getValGustativa() { return valGustativa;   }
   public void setValGustativa(String valGustativa) {  this.valGustativa = valGustativa;   }
   public String getValVisual() { return valVisual;   }
   public void setValVisual(String valVisual) {  this.valVisual = valVisual;   }
```

```
    public int getNota() {  return nota;    }
    public void setNota(int nota) {  this.nota = nota;    }
    public String getFoto() {  return foto;    }
    public void setFoto(String foto) {  this.foto = foto;    }
    public String getTipus() { return tipus;    }
    public void setTipus(String tipus) {  this.tipus = tipus;    }
}
```

Crearem tantes classes com taules tingui el nostre projecte (bodega,denominació el nostre cas)

Crearem una classe per crear, obrir, tancar i controlar els canvis d'estructura de la base de dades a partir de la classe SQLiteOpenHelper.
Com a bona pràctica ens convé crear un conjunt de constants amb els noms de les taules i columnes de la nostra base de dades, aprofitarem la nova classe per posar-hi les constants encara que ho podem fer a un altre lloc (una classe apart per exemple.

```
public class HelperVi extends SQLiteOpenHelper {
    public static final String TABLE_VI = "vi";
    public static final String COLUMN_ID = "_id";
    public static final String COLUMN_NOMVI = "nomvi";
    public static final String COLUMN_ANADA = "anada";
    public static final String COLUMN_TIPUS = "tipus";
    public static final String COLUMN_LLOC = "lloc";
    public static final String COLUMN_GRADUACIO = "graduacio";
    public static final String COLUMN_DATA = "data";
    public static final String COLUMN_COMENTARI = "comentari";
    public static final String COLUMN_IDBODEGA = "idbodega";
    public static final String COLUMN_IDDENOMINACIO = "iddenominacio";
    public static final String COLUMN_PREU = "preu";
    public static final String COLUMN_VALOLFATIVA = "valolfativa";
    public static final String COLUMN_VALGUSTATIVA = "valgustativa";
    public static final String COLUMN_VALVISUAL = "valvisual";
    public static final String COLUMN_NOTA = "nota";
    public static final String COLUMN_FOTO = "foto";
    public static final String TABLE_BODEGA = "bodega";
    public static final String COLUMN__IDBODEGA = "_idbodega";
    public static final String COLUMN_NOMBODEGA = "nombodega";
    public static final String TABLE_DENOMINACIO = "denominacio";
    public static final String COLUMN__IDDENOMINACIO = "_iddenominacio";
    public static final String COLUMN_NOMDENOMINACIO = "nomdenominacio";
    public static final String TABLE_TIPUS = "tipus";
    public static final String COLUMN__TIPUS = "tipus";
    private static final String DATABASE_NAME = "wineapp";
    private static final int DATABASE_VERSION = 1;      // Controla la versió de la base de dades

    // Setències de creació de la base de dades
    private static final String DATABASE_CREATE_VI = "create table "
          + TABLE_VI + "(" + COLUMN_ID
          + " integer primary key autoincrement, "
          + COLUMN_NOMVI + " text not null,"
          + COLUMN_ANADA + " text,"
          + COLUMN_TIPUS + " text,"
          + COLUMN_LLOC + " text,"
          + COLUMN_GRADUACIO + " text,"
          + COLUMN_DATA + " text,"
          + COLUMN_COMENTARI + " text,"
          + COLUMN_IDBODEGA + " integer,"
          + COLUMN_IDDENOMINACIO + " integer,"
          + COLUMN_PREU + " float,"
          + COLUMN_VALOLFATIVA + " text,"
          + COLUMN_VALGUSTATIVA + " text,"
          + COLUMN_VALVISUAL + " text,"
          + COLUMN_NOTA + " integer,"
          + COLUMN_FOTO + " text);";
    private static final String DATABASE_CREATE_BODEGA = "create table "
          + TABLE_BODEGA + "("
          + COLUMN__IDBODEGA + " integer primary key autoincrement, "
```

```
        + COLUMN_NOMBODEGA + " text not null);";
    private static final String DATABASE_CREATE_DENOMINACIO = "create table "
        + TABLE_DENOMINACIO + "("
        + COLUMN__IDDENOMINACIO + " integer primary key autoincrement, "
        + COLUMN_NOMDENOMINACIO + " text not null);";
    private static final String DATABASE_CREATE_TIPUS = "create table "
        + TABLE_TIPUS + "("
        + COLUMN__TIPUS + " text not null primary key);";

    public HelperVi(Context context) {
      super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }

    @Override
    public void onCreate(SQLiteDatabase database) {
      // CREAM LA BASE DE DADES
      database.execSQL(DATABASE_CREATE_VI);
      database.execSQL(DATABASE_CREATE_BODEGA);
      database.execSQL(DATABASE_CREATE_DENOMINACIO);
      database.execSQL(DATABASE_CREATE_TIPUS);
      database.execSQL(" insert into "+TABLE_TIPUS+"(tipus) values(('Tinto'))");
      database.execSQL(" insert into "+TABLE_TIPUS+"(tipus) values(('Rosat'))");
      database.execSQL(" insert into "+TABLE_TIPUS+"(tipus) values(('Blanc'))");
      database.execSQL(" insert into "+TABLE_TIPUS+"(tipus) values(('Dolç'))");
      database.execSQL(" insert into "+TABLE_TIPUS+"(tipus) values(('Espumós'))");
      database.execSQL(" insert into "+TABLE_TIPUS+"(tipus) values(('Cervesa'))");
      database.execSQL(" insert into "+TABLE_TIPUS+"(tipus) values(('Altres'))");
    }
    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
      // Detecta si hi ha una canvi a  DATABASE_VERSION i recrea la base de dades
      Log.w(HelperVi.class.getName(),
            "Modificant desde la versió " + oldVersion + " a "+ newVersion );
      db.execSQL("DROP TABLE IF EXISTS " + TABLE_DENOMINACIO);
      db.execSQL("DROP TABLE IF EXISTS " + TABLE_BODEGA);
      db.execSQL("DROP TABLE IF EXISTS " + TABLE_TIPUS);
      db.execSQL("DROP TABLE IF EXISTS " + TABLE_VI);
      onCreate(db);
    }
}
```

Una vegada creada la classe per obrir,crear i recrear la base de dades crearem una nova classe per implementar els mètodes necessaris per intereactuar amb la base de dades amb l'ajuda de la classe **SQLiteDatabase**.

```
public class DataSourceVi {

    private SQLiteDatabase database;
    private HelperVi dbAjuda;                   //CLASSE AJUDA

    private String[] allColumnsVi = { HelperVi.COLUMN_ID,HelperVi.COLUMN_NOMVI,HelperVi.COLUMN_ANADA,
                        HelperVi.COLUMN_LLOC, HelperVi.COLUMN_GRADUACIO,HelperVi.COLUMN_DATA,
                        HelperVi.COLUMN_COMENTARI,HelperVi.COLUMN_IDBODEGA,
                         HelperVi.COLUMN_IDDENOMINACIO,HelperVi.COLUMN_PREU,
                        HelperVi.COLUMN_VALOLFATIVA,HelperVi.COLUMN_VALGUSTATIVA,
                         HelperVi.COLUMN_VALVISUAL,HelperVi.COLUMN_NOTA,HelperVi.COLUMN_FOTO,
                        HelperVi.COLUMN_TIPUS};

    public DataSourceVi(Context context) {  //CONSTRUCTOR
      dbAjuda = new HelperVi(context);
    }

    public void open() throws SQLException {
      database = dbAjuda.getWritableDatabase();
    }
    public void close() {
      dbAjuda.close();
    }
```

```
public Vi createVi(Vi vi) {
    // insert d'un nou vi
    ContentValues values = new ContentValues();
    values.put(HelperVi.COLUMN_NOMVI, vi.getNomVi());
    values.put(HelperVi.COLUMN_ANADA, vi.getAnada());
    values.put(HelperVi.COLUMN_TIPUS, vi.getTipus());
    values.put(HelperVi.COLUMN_LLOC, vi.getLloc());
    values.put(HelperVi.COLUMN_GRADUACIO, vi.getGraduacio());
    values.put(HelperVi.COLUMN_DATA, String.valueOf(vi.getData()));
    values.put(HelperVi.COLUMN_COMENTARI, vi.getComentari());
    values.put(HelperVi.COLUMN_IDBODEGA, vi.getIdBodega());
    values.put(HelperVi.COLUMN_IDDENOMINACIO,vi.getIdDenominacio());
    values.put(HelperVi.COLUMN_PREU,vi.getPreu());
    values.put(HelperVi.COLUMN_VALOLFATIVA,vi.getValOlfativa());
    values.put(HelperVi.COLUMN_VALGUSTATIVA,vi.getValGustativa());
    values.put(HelperVi.COLUMN_VALVISUAL,vi.getValVisual());
    values.put(HelperVi.COLUMN_NOTA,vi.getNota() );
    values.put(HelperVi.COLUMN_FOTO,vi.getFoto() );
    long insertId = database.insert(HelperVi.TABLE_VI, null,values);
    vi.setId(insertId);
    return vi;
}

public boolean updateVi(Vi vi) {
    // update vi
    ContentValues values = new ContentValues();
    long id=vi.getId();
    values.put(HelperVi.COLUMN_NOMVI, vi.getNomVi());
    values.put(HelperVi.COLUMN_ANADA, vi.getAnada());
    values.put(HelperVi.COLUMN_LLOC, vi.getLloc());
    values.put(HelperVi.COLUMN_TIPUS, vi.getTipus());
    values.put(HelperVi.COLUMN_GRADUACIO, vi.getGraduacio());
    values.put(HelperVi.COLUMN_DATA, String.valueOf(vi.getData()));
    values.put(HelperVi.COLUMN_COMENTARI, vi.getComentari());
    values.put(HelperVi.COLUMN_IDBODEGA, vi.getIdBodega());
    values.put(HelperVi.COLUMN_IDDENOMINACIO,vi.getIdDenominacio());
    values.put(HelperVi.COLUMN_PREU,vi.getPreu());
    values.put(HelperVi.COLUMN_VALOLFATIVA,vi.getValOlfativa());
    values.put(HelperVi.COLUMN_VALGUSTATIVA,vi.getValGustativa());
    values.put(HelperVi.COLUMN_VALVISUAL,vi.getValVisual());
    values.put(HelperVi.COLUMN_NOTA,vi.getNota() );
    values.put(HelperVi.COLUMN_FOTO,vi.getFoto() );
    return database.update(HelperVi.TABLE_VI, values,HelperVi.COLUMN_ID + "=" + id, null) > 0;
}

public void deleteVi(Vi vi) {
    long id = vi.getId();
    database.delete(HelperVi.TABLE_VI, HelperVi.COLUMN_ID + " = " + id, null);
}

public Vi getVi(long id) {
    Vi vi;
    Cursor cursor = database.query(HelperVi.TABLE_VI,
            allColumnsVi, HelperVi.COLUMN_ID + " = " + id, null,
            null, null, null);
    if (cursor.getCount()>0) {
        cursor.moveToFirst();
        vi = cursorToVi(cursor);
    } else { vi=new Vi(); }    // id=-1 no trobat
     cursor.close();
    return vi;
}

public List<Vi> getAllVi() {
    List<Vi> vins = new ArrayList<Vi>();
    Cursor cursor = database.query(HelperVi.TABLE_VI,allColumnsVi, null, null, null, null,
                                                HelperVi.COLUMN_DATA+" DESC");
    cursor.moveToFirst();
    while (!cursor.isAfterLast()) {
```

```
        Vi vi = cursorToVi(cursor);
        vins.add(vi);
        cursor.moveToNext();
    }
    // Make sure to close the cursor
    cursor.close();
    return vins;
}

private Vi cursorToVi(Cursor cursor) {
    Vi v = new Vi();
    v.setId(cursor.getLong(0));
    v.setNomVi(cursor.getString(1));
    v.setAnada(cursor.getString(2));
    v.setLloc(cursor.getString(3));
    v.setGraduacio(cursor.getString(4));
    v.setData(cursor.getString(5));
    v.setComentari(cursor.getString(6));
    v.setIdBodega(cursor.getLong(7));
    v.setIdBodega(cursor.getLong(8));
    v.setPreu(cursor.getFloat(9));
    v.setValOlfativa(cursor.getString(10));
    v.setValGustativa(cursor.getString(11));
    v.setValVisual(cursor.getString(12));
    v.setNota(cursor.getInt(13));
    v.setFoto(cursor.getString(14));
    return v;
}

//CREAREM EL MÈTODES QUE ENS FACIN FALTA EN FUNCIÓ DE LA NOSTRA BASE DE DADES

 .......
```

Ara necessitarem implementar com a mínim dues activities, una activitat principal **MainActivity** que pot esser un ListView per mostrar una llista de les tuples de la nostra taula principal (Ex:vins) i una activitat per editar o modificar un nou element que podem anomenar **EditActivity**.c

La nostra activitat principal contindrà un ListView i carregarà els elements de la base de dades, per fer això podem fer servir un adaptador del tipus **SimpleAdapter** o ho podem fer de la mateixa manera que ho feram al exercici del ***CustomerView*** fent el nostre «adapter» personalitzat. També contindrà un botó per afegir un nou element (vi) a la taula.

```
public class MainActivity extends ListActivity {

  private ListAdapter adapter;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    ListView lv = getListView();

    lv.setOnItemClickListener(new AdapterView.OnItemClickListener() {
      // En fer onClick a un element de la llista cridam l'activity d'edició i passam la clau primària
      @Override
      public void onItemClick(AdapterView<?> parent, View view,
                              int position, long id) {
        String s = ((TextView) view.findViewById(R.id.id)).getText().toString();
        Intent in=new Intent(getApplicationContext(),EditaVi.class);
        in.putExtra("ID", s);
        startActivity(in);
      }
    });
```

```
        Button btNou=(Button) findViewById(R.id.nouBtn);
        btNou.setOnClickListener(
                // Cridam l'activity d'edició indicant que es un insert (clau primària en blanc per exemple)
                new View.OnClickListener() {
                    @Override
                    public void onClick(View v) {
                        Intent in=new Intent(getApplicationContext(),EditaVi.class);
                        in.putExtra("ID", "");
                        startActivity(in);
                    }
                }
        );
        mostraVins();   // Carrega la llista
    }

    public void mostraVins() {
        // Obrim la base de dades
        DataSourceVi bd;
        bd = new DataSourceVi(this);
        bd.open();

        // Obtenim tots els vins
        List<Vi> llistaVins = bd.getAllVi();
        ArrayList<HashMap<String, String>> llista = new ArrayList<HashMap<String, String>>();
        for (int i = 0; i < llistaVins.size(); i++) {
            HashMap<String, String> map = new HashMap<String, String>();
            Vi vi = llistaVins.get(i);
            map.put("id", String.valueOf(vi.getId()));
            map.put("nomVi", vi.getNomVi());
            map.put("data", vi.getData());
            map.put("tipus", vi.getTipus());
            llista.add(map);
        }

        //Tanquem la BD
        bd.close();

        //Assignar a la listview
        adapter = new SimpleAdapter(this, llista,R.layout.llistavins,
                    new String[]{"id", "nomVi", "data", "tipus"},
                    new int[]{R.id.id, R.id.nomVi, R.id.data, R.id.tipus});
        setListAdapter(adapter);
    }

}
```

A l'activity per l'edició crearem una interfície d'acord amb les nostres necessitats i farem un insert o update de les dades per exemple amb un botó d'ok.

En tornar a l'activitat principal hem de recarregar la llista (aquesta s'ha modificat), això ho podem fer al mètode de callback OnResume:

```
@Override
public void onResume() {
    super.onResume();
    mostraVins();
}
```