Step-1: Design………. For 3 fields, id, name, qty

Step-2:

public class Product {

private int id;

private String productname;

private int quantity;}

Step-3:

public class MyDbAdapter {

MyDbHelper helper;

public MyDbAdapter(Context context) {

this.helper = new MyDbHelper(context);

}

static class MyDbHelper extends SQLiteOpenHelper {

// Database Information

static final String DB\_NAME = "PROD.DB";

// database version

static final int DB\_VERSION = 1;

public MyDbHelper(Context context) {

super(context, DB\_NAME, null, DB\_VERSION);

}

////////// Start ///////////

// Table Name

public static final String TABLE\_NAME = "PRODUCT";

// Table columns

public static final String ID = "id";

private static final String PRODUCT\_NAME = "product\_name";

private static final String QTY = "qty";

// Creating table query

private static final String CREATE\_TABLE = "create table " + TABLE\_NAME + "(" + ID

+ " INTEGER PRIMARY KEY AUTOINCREMENT, " + PRODUCT\_NAME + " TEXT NOT NULL, " + QTY + " INTEGER NOT NULL)";

@Override

public void onCreate(SQLiteDatabase db) {

db.execSQL(CREATE\_TABLE);

}

@Override

public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {

db.execSQL("DROP TABLE IF EXISTS " + TABLE\_NAME);

onCreate(db);

}

}

public long insertData(Product product) {

SQLiteDatabase db = helper.getWritableDatabase();

ContentValues cv = new ContentValues();

cv.put(MyDbHelper.PRODUCT\_NAME, product.getProductname());

cv.put(MyDbHelper.QTY, product.getQuantity());

long id = db.insert(MyDbHelper.TABLE\_NAME, null, cv);

return id;

}

public List<Product> getList() {

SQLiteDatabase db = helper.getReadableDatabase();

String[] projection = {MyDbHelper.ID,

MyDbHelper.PRODUCT\_NAME, MyDbHelper.QTY};

Cursor cursor = db.query(

MyDbHelper.TABLE\_NAME, // The table to query

projection, // The array of columns to return (pass null to get all)

null, // The columns for the WHERE clause

null, // The values for the WHERE clause

null, // don't group the rows

null, // don't filter by row groups

null // The sort order

);

List<Product> list = new ArrayList<>();

while (cursor.moveToNext()) {

Product product = new Product(Integer.parseInt(cursor.getString(0)), cursor.getString(1), Integer.parseInt(cursor.getString(2)));

list.add(product);

}

cursor.close();

return list;

}

public void deleteProduct(int id) {

SQLiteDatabase db = helper.getWritableDatabase();

db.execSQL("DELETE FROM " + MyDbHelper.TABLE\_NAME + " WHERE " + MyDbHelper.ID + "='" + id + "'");

db.close();

}

public Product findProductById(int id) {

SQLiteDatabase db = helper.getReadableDatabase();

String[] projection = {MyDbHelper.ID,

MyDbHelper.PRODUCT\_NAME, MyDbHelper.QTY};

String selection = MyDbHelper.ID + " = " + id;

Cursor cursor = db.query(

MyDbHelper.TABLE\_NAME, // The table to query

projection, // The array of columns to return (pass null to get all)

selection, // The columns for the WHERE clause

null, // The values for the WHERE clause

null, // don't group the rows

null,

null

);

Product product = new Product();

if (cursor.moveToFirst()) {

cursor.moveToFirst();

product.setId(Integer.parseInt(cursor.getString(0)));

product.setProductname(cursor.getString(1));

product.setQuantity(Integer.parseInt(cursor.getString(2)));

cursor.close();

} else {

product = null;

}

return product;

}}

Step-5:

public class Message {

public static void message(Context context, String message) {

Toast.makeText(context, message, Toast.LENGTH\_LONG).show();

}

}

Step-6:

public class MainActivity extends AppCompatActivity {

EditText id, name, qty;

MyDbAdapter helper;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

id = (EditText) findViewById(R.id.productID);

name = (EditText) findViewById(R.id.productName);

qty = (EditText) findViewById(R.id.productQuantity);

helper = new MyDbAdapter(this);

getProductlist();

}

public void saveProduct(View view) {

Product product = new Product(name.getText().toString(), Integer.parseInt(qty.getText().toString()));

long i = helper.insertData(product);

if (i < 0) {

Message.message(this, "Unsuccessful");

} else {

Message.message(this, "Successful");

}

}

public void getProductByProductId(View view){

int pid=Integer.parseInt(id.getText().toString().trim());

Product p=helper.findProductById(pid);

if(p != null) {

name.setText(p.getProductname());

qty.setText(String.valueOf(p.getQuantity()));

}else{

Toast.makeText(this, "No Data Exists", Toast.LENGTH\_SHORT).show();

}

}

public void deleteProductByProductId(View view){

int pid=Integer.parseInt(id.getText().toString().trim());

helper.deleteProduct(pid);

Toast.makeText(this, "Success", Toast.LENGTH\_SHORT).show();

}

public void getProductlist(){

List<Product> p=helper.getList();

System.out.println(p.size());

}

}