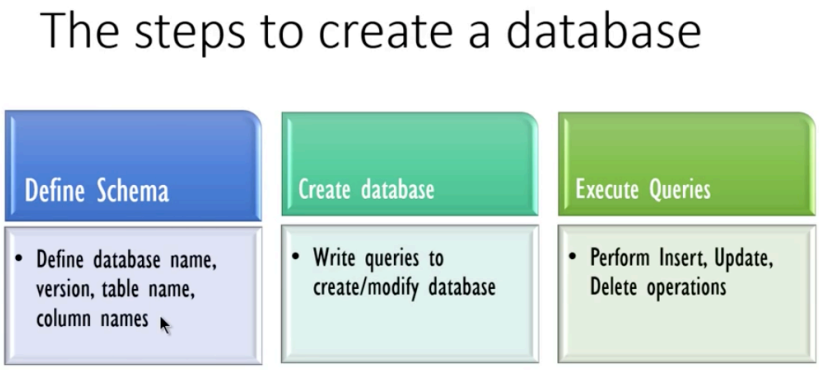
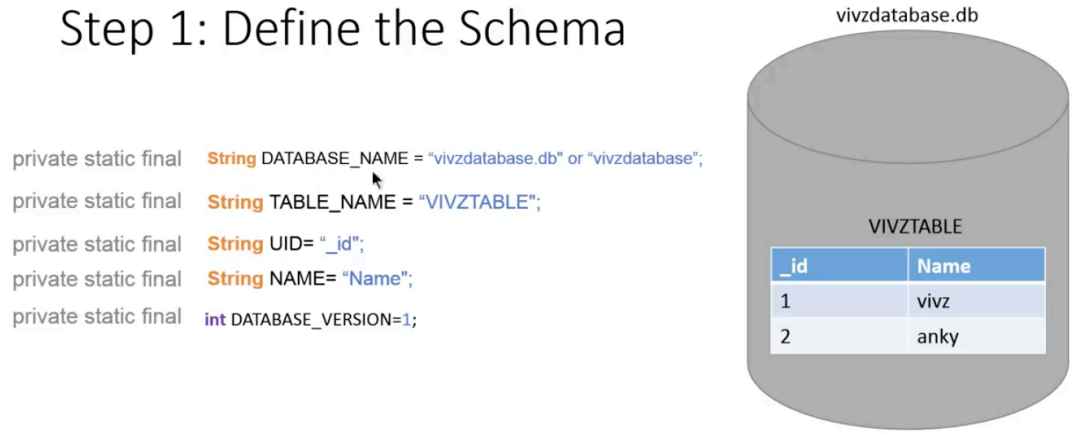
SQLite Database Schema



In Android, it is a common convention that primary keys start with underscore. i.e. "_id".



The SQLiteOpenHelper

The SQLiteOpenHelper

- Create a subclass of SQLiteOpenHelper implementing onCreate(SQLiteDatabase),onUpgrade(SQLiteDatabase, int, int)
- This class takes care of opening the database if it exists, creating it if it does not, and upgrading it as necessary.


```
class VivzHelper extends SQLiteOpenHelper
{
    VivzHelper(Context context)
    {
       super(context,  DATABASE_NAME,  null,  DATABASE_VERSION  );
    }
    public void onCreate(SQLiteDatabase db)
    {
    }
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion)
    {
    }
}
```


The SQLiteOpenHelper

- **onCreate():** Called when the database is created for the first time. Creation of tables and initial data inside tables should be put here.
- **onUpgrade():** Called when the database needs to be upgraded. Use this method to drop tables, add tables, or do anything else it needs to upgrade to the new schema version.
- If you add new columns you can use ALTER TABLE to insert them into a live table.
- If you rename or remove columns you can use ALTER TABLE to rename the old table, then create the new table and then populate the new table with the contents of the old table.


```
class VivzHelper extends SQLiteOpenHelper
{
    VivzHelper(Context context)
    {
       super(context,  DATABASE_NAME,  null,  DATABASE_VERSION  );
    }
    public void onCreate(SQLiteDatabase db)
    {
    }
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion)
    {
    }
}
```


SQLite Database class

The SQLiteDatabase class

- SQLiteDatabase has methods to create, delete, execute SQL commands, and perform other common database management tasks.
- Database names must be unique within an application, not across all applications.
- **public void execSQL(String sql)**
- Execute a single SQL statement that is NOT a SELECT or any other SQL statement that returns data.
- Multiple statements separated by semicolons are not supported.
- If the SQL string is invalid, throws an SQLException

Subscribe Here

Step1:Implement the onCreate

Step1: Implement the onCreate

```
class VivzHelper extends SQLiteOpenHelper
{
    private static final  String DATABASE_NAME = "vivzdatabase.db";
    private static final  String TABLE_NAME = "VIVZTABLE";
    private static final  String UID= "_id";
    private static final  String NAME= "Name";
    private static final  int DATABASE_VERSION=1;
    VivzHelper(Context context)
    {
      super ( context,  DATABASE_NAME,  null,  DATABASE_VERSION    );
    }
    public void onCreate(SQLiteDatabase db)
    {
     try{
     db.execSQL( "CREATE TABLE VIVZTABLE (_id INTEGER PRIMARY KEY
                  AUTOINCREMENT, Name VARCHAR(255));" );
      }catch(SQLException e){
        e.printStackTrace();
      }
    }
     public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion)
     {
     }
}
```

Step1: Implement the onUpgrade

Items on the update are flexible depending on what users want to do. Options includes drop table, save table content, add table.

Step1: Implement the onUpgrade

```
class VivzHelper extends SQLiteOpenHelper
{
    private static final   String DATABASE_NAME = "vivzdatabase.db";
    private static final   String TABLE_NAME = "VIVZTABLE";
    private static final   String UID= "_id";
    private static final   String NAME= "Name";
    private static final   int DATABASE_VERSION=1;
    VivzHelper(Context context)
    {
      super  (  context,   DATABASE_NAME,    null,  DATABASE_VERSION     );
    }
    public void onCreate(SQLiteDatabase db)
    {
       ...
    }
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion)
    {

        db.execSQL("DROP TABLE IF EXISTS VIVZTABLE");

        onCreate(db);

    }
}
```

OnCreate is called again to recreate the table with the new statement that we added