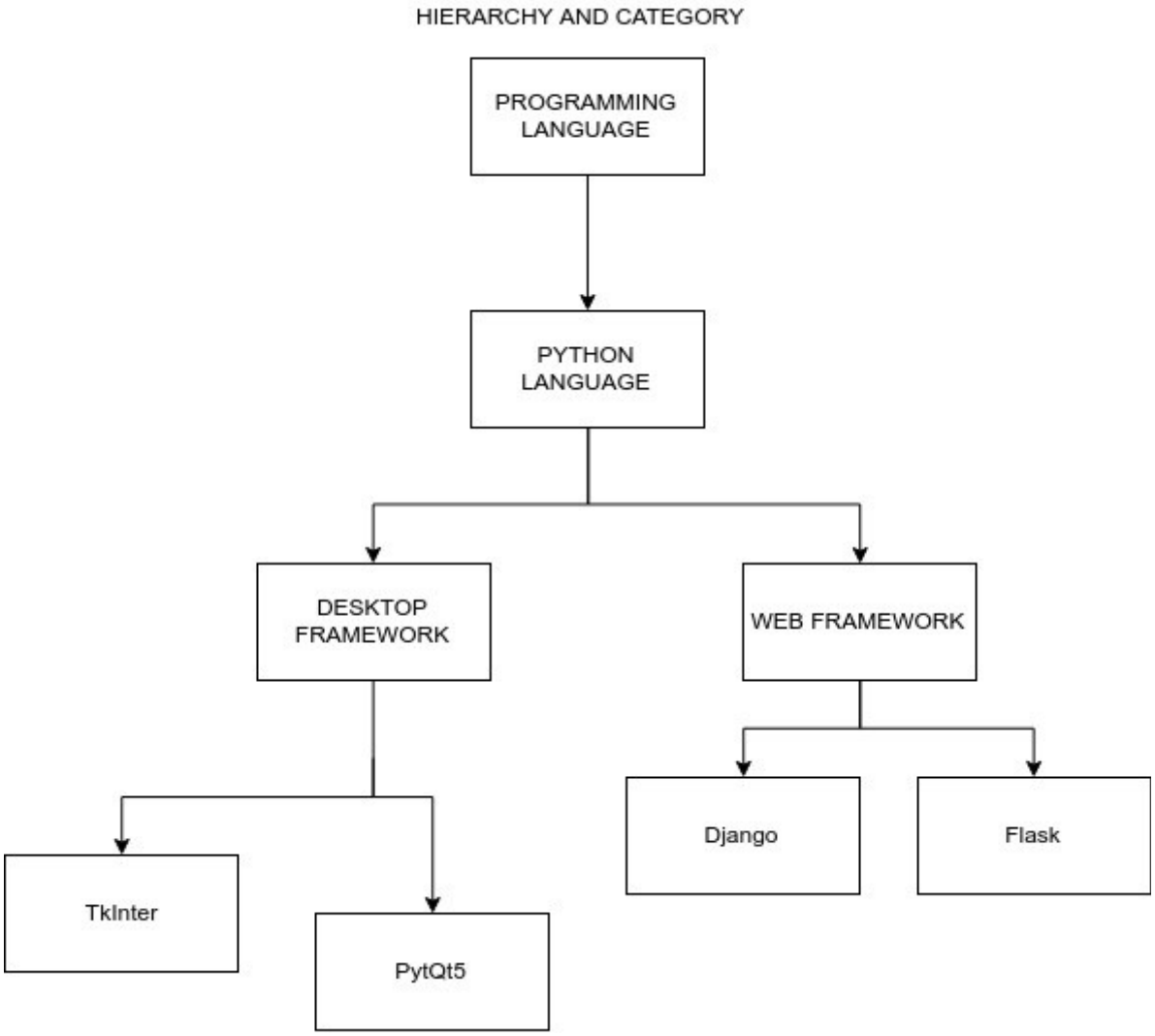
TUGAS BASIS DATA
TREE MODEL STRUCTURE



Alfian Tegar Putra Afandi
17050623015 / D3 – Mi 2017

Model Tree Structures with Parent References :

- Create db:

```
> db
tugas_basdat2
```

- Insert this db:

```
> db.category.insert({ _id:"tkinter", parent:"desktop" })
WriteResult({ "nInserted" : 1 })
> db.category.insert({ _id:"pyqt5", parent:"desktop" })
WriteResult({ "nInserted" : 1 })
> db.category.insert({ _id:"django", parent:"web" })
WriteResult({ "nInserted" : 1 })
> db.category.insert({ _id:"flask", parent:"web" })
WriteResult({ "nInserted" : 1 })
> db.category.insert({ _id:"desktop", parent:"pythonLanguage" })
WriteResult({ "nInserted" : 1 })
> db.category.insert({ _id:"web", parent:"pythonLanguage" })
WriteResult({ "nInserted" : 1 })
> db.category.insert({ _id:"pythonLanguage", parent:"programmingLanguage" })
WriteResult({ "nInserted" : 1 })
> db.category.insert({ _id:"programmingLanguage", parent:null })
WriteResult({ "nInserted" : 1 })
> _
```

- Look for one of the data :

```
> db.category.findOne({ _id:"django" }).parent
web
> db.category.findOne({ _id:"tkinter" }).parent
desktop
> _
```

- find parent :

```
> db.category.find({parent:"web"})
{ "_id" : "django", "parent" : "web" }
{ "_id" : "flask", "parent" : "web" }
> _
```

Model Tree Structures with Child References :

```
> use treedb
switched to db treedb
> db
treedb
> db.cat.insert({ _id:"django", children:[] })
WriteResult({ "nInserted" : 1 })
> db.cat.insert({ _id:"flask", children:[] })
WriteResult({ "nInserted" : 1 })
> db.cat.insert({ _id:"webFramework", children:["django","flask"] })
WriteResult({ "nInserted" : 1 })
> db.cat.insert({ _id:"tkinter", children:[] })
WriteResult({ "nInserted" : 1 })
> db.cat.insert({ _id:"pyqt5", children:[] })
WriteResult({ "nInserted" : 1 })
> db.cat.insert({ _id:"desktopFramework", children:["tkinter","pyqt5"] })
WriteResult({ "nInserted" : 1 })
> db.cat.insert({ _id:"pythonLanguage", children:["webFramework","desktopFramework"] })
WriteResult({ "nInserted" : 1 })
> db.cat.insert({ _id:"programmingLanguage", children:["pythonLanguage"] })
WriteResult({ "nInserted" : 1 })
>
>
> # find one
2019-03-27T23:10:06.357+0700 E QUERY    [js] SyntaxError: illegal character @(shell):1:0
> db.cat.findOne({ _id:"webFramework" }).children
[ "django", "flask" ]
>
>
> # create index #
2019-03-27T23:11:13.144+0700 E QUERY    [js] SyntaxError: illegal character @(shell):1:0
> db.cat.createIndex({ children:1 })
{
        "createdCollectionAutomatically" : false,
        "numIndexesBefore" : 1,
        "numIndexesAfter" : 2,
        "ok" : 1
}
>
>
> # find parent
2019-03-27T23:11:47.625+0700 E QUERY    [js] SyntaxError: illegal character @(shell):1:0
>
> db.cat.find({ children:"desktopFramework" })
{ "_id" : "pythonLanguage", "children" : [ "webFramework", "desktopFramework" ] }
> _
```

Model Tree Structures with an Array of Ancestors :

```
> use arraydb
switched to db arraydb
> db
arraydb
> db.cat.insert({ _id:"django", ancestors:["programmingLanguage","pythonLanguage","webFramework"], parent:"webFramework" })
WriteResult({ "nInserted" : 1 })
> db.cat.insert({ _id:"flask", ancestors:["programmingLanguage","pythonLanguage","webFramework"], parent:"webFramework" })
WriteResult({ "nInserted" : 1 })
> db.cat.insert({ _id:"tkinter", ancestors:["programmingLanguage","pythonLanguage","desktopFramework"], parent:"desktopFramework" })
WriteResult({ "nInserted" : 1 })
> db.cat.insert({ _id:"pyqt5", ancestors:["programmingLanguage","pythonLanguage","desktopFramework"], parent:"desktopFramework" })
WriteResult({ "nInserted" : 1 })
> db.cat.insert({ _id:"webFramework", ancestors:["programmingLanguage","pythonLanguage"], parent:"pythonLanguage" })
WriteResult({ "nInserted" : 1 })
> db.cat.insert({ _id:"desktopFramework", ancestors:["programmingLanguage","pythonLanguage"], parent:"pythonLanguage" })
WriteResult({ "nInserted" : 1 })
> db.cat.insert({ _id:"pythonLanguage", ancestors:["programmingLanguage"], parent:"programmingLanguage" })
WriteResult({ "nInserted" : 1 })
> db.cat.insert({ _id:"programmingLanguage", ancestors:[], parent:null })
WriteResult({ "nInserted" : 1 })
>
>
> # findOne
2019-03-27T23:21:04.513+0700 E QUERY    [js] SyntaxError: illegal character @(shell):1:0
>
> db.cat.findOne({ _id:"flask" }).ancestors
[ "programmingLanguage", "pythonLanguage", "webFramework" ]
>
>
> #create index
2019-03-27T23:21:48.395+0700 E QUERY    [js] SyntaxError: illegal character @(shell):1:0
>
> db.cat.createIndex({ ancestors:1 })
{
        "createdCollectionAutomatically" : false,
        "numIndexesBefore" : 1,
        "numIndexesAfter" : 2,
        "ok" : 1
}
>
>
> db.cat.find({ ancestors:"webFramework" })
{ "_id" : "django", "ancestors" : [ "programmingLanguage", "pythonLanguage", "webFramework" ], "parent" : "webFramework" }
{ "_id" : "flask", "ancestors" : [ "programmingLanguage", "pythonLanguage", "webFramework" ], "parent" : "webFramework" }
> _
```

Model Tree Structures with Materialized Paths :

```
> use materialized
switched to db materialized
> db
materialized
> db.cat.insert({ _id:"programmingLanguage", path:null })
WriteResult({ "nInserted" : 1 })
> db.cat.insert({ _id:"pythonLanguage", path:",programmingLanguage," })
WriteResult({ "nInserted" : 1 })
> db.cat.insert({ _id:"webFramework", path:",programmingLanguage,pythonLanguage," })
WriteResult({ "nInserted" : 1 })
> db.cat.insert({ _id:"desktopFramework", path:",programmingLanguage,pythonLanguage," })
WriteResult({ "nInserted" : 1 })
> db.cat.insert({ _id:"tkinter", path:",programmingLanguage,pythonLanguage,desktopFramework," })
WriteResult({ "nInserted" : 1 })
> db.cat.insert({ _id:"pyqt5", path:",programmingLanguage,pythonLanguage,desktopFramework," })
WriteResult({ "nInserted" : 1 })
> db.cat.insert({ _id:"django", path:",programmingLanguage,pythonLanguage,webFramework," })
WriteResult({ "nInserted" : 1 })
> db.cat.insert({ _id:"flask", path:",programmingLanguage,pythonLanguage,webFramework," })
WriteResult({ "nInserted" : 1 })
>
>
> db.cat.find().sort({ path:1 })
{ "_id" : "programmingLanguage", "path" : null }
{ "_id" : "pythonLanguage", "path" : ",programmingLanguage," }
{ "_id" : "webFramework", "path" : ",programmingLanguage,pythonLanguage," }
{ "_id" : "desktopFramework", "path" : ",programmingLanguage,pythonLanguage," }
{ "_id" : "tkinter", "path" : ",programmingLanguage,pythonLanguage,desktopFramework," }
{ "_id" : "pyqt5", "path" : ",programmingLanguage,pythonLanguage,desktopFramework," }
{ "_id" : "django", "path" : ",programmingLanguage,pythonLanguage,webFramework," }
{ "_id" : "flask", "path" : ",programmingLanguage,pythonLanguage,webFramework," }
>
>
> db.cat.find({ path:/,webFramework,/ })
{ "_id" : "django", "path" : ",programmingLanguage,pythonLanguage,webFramework," }
{ "_id" : "flask", "path" : ",programmingLanguage,pythonLanguage,webFramework," }
>
>
> db.cat.createIndex({path:1})
{
        "createdCollectionAutomatically" : false,
        "numIndexesBefore" : 1,
        "numIndexesAfter" : 2,
        "ok" : 1
}
> _
```