Nama : Widyatna Dahniah

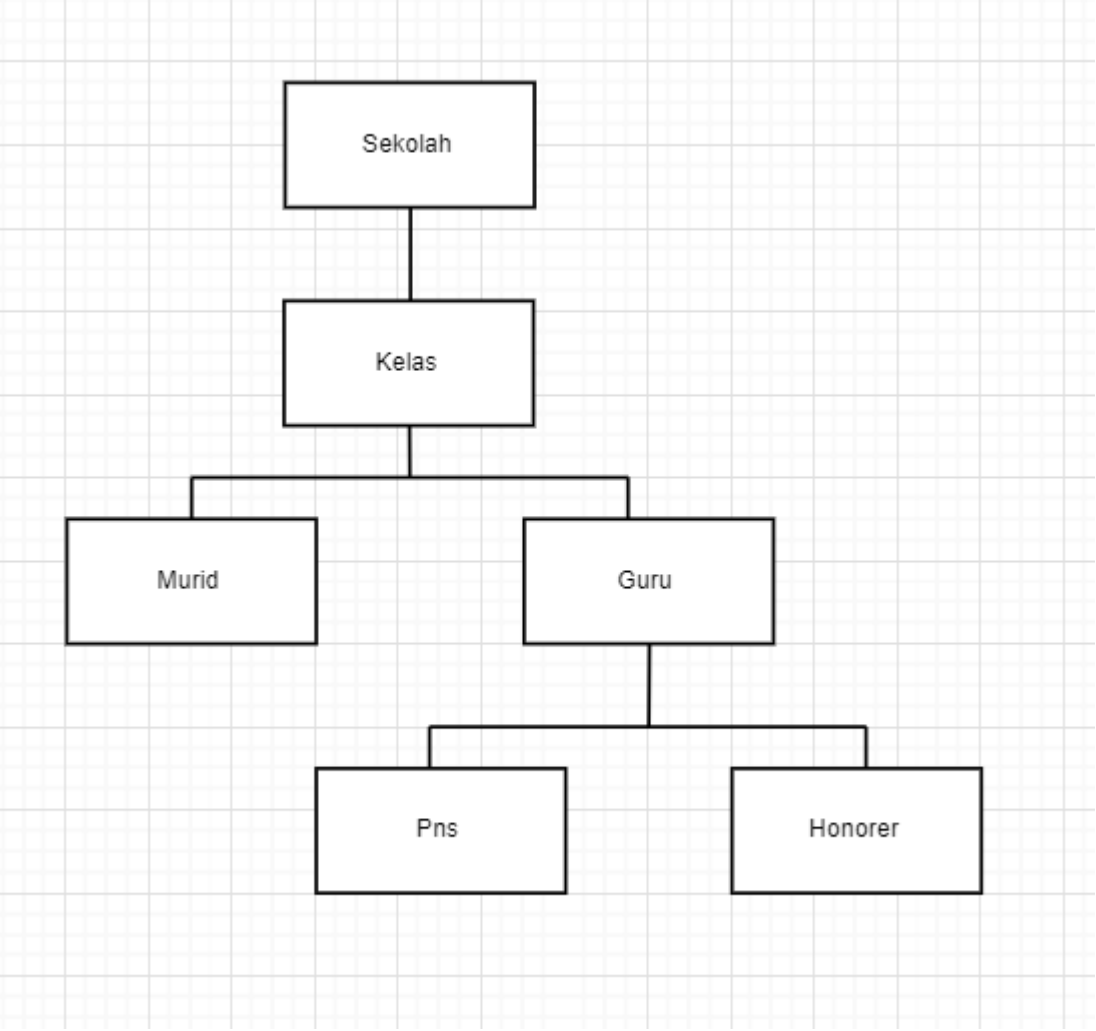
NIM : 17050623025

Prodi : D3 Manajemen Informatika

TUGAS BASIS DATA LANJUT

MONGO DB TREE STRUCTURE

Tingkatan Field dari Collection "Kategori"



1. Parrent References

```
> show databases
admin            0.000GB
config           0.000GB
hutanrimbun      0.000GB
local            0.000GB
parentreferences  0.000GB
> use parent_preferences
switched to db parent_preferences
> db.categories.insert({_id:'Pns', parent:'Guru'})
WriteResult({ "nInserted" : 1 })
> db.categories.insert({_id:'Honorer', parent:'Guru'})
WriteResult({ "nInserted" : 1 })
> db.categories.insert({_id:'Guru', parent:'Kelas'})
WriteResult({ "nInserted" : 1 })
> db.categories.insert({_id:'Murid', parent:'Kelas'})
WriteResult({ "nInserted" : 1 })
> db.categories.insert({_id:'Kelas', parent:'Sekolah'})
WriteResult({ "nInserted" : 1 })
> db.categories.insert({_id:'Kelas', parent:null})
WriteResult({
        "nInserted" : 0,
        "writeError" : {
                "code" : 11000,
                "errmsg" : "E11000 duplicate key error collection: parent_preferences.categories index: _id_ dup key: { : \"Kelas\" }"
        }
})
> db.categories.insert({_id:'Sekolah', parent:null})
WriteResult({ "nInserted" : 1 })
```

```
> db.categories.findOne({_id:'Pns'}).parent
Guru
```

```
> db.categories.createIndex({parent:1})
{
        "createdCollectionAutomatically" : false,
        "numIndexesBefore" : 1,
        "numIndexesAfter" : 2,
        "ok" : 1
```

```
> db.categories.find({parent:'Guru'})
{ "_id" : "Pns", "parent" : "Guru" }
{ "_id" : "Honorer", "parent" : "Guru" }
>
```

2. Child References

C:\WINDOWS\system32\cmd.exe - mongo

```
> show databases
admin               0.000GB
config              0.000GB
hutanrimbun         0.000GB
local               0.000GB
parent_preferences  0.000GB
parentreferences    0.000GB
> use childpreferences
switched to db childpreferences
> db.categories.insert({_id:'Pns', children:[]})
Display all 180 possibilities? (y or n)
> db.categories.insert({_id:'Pns', children:[]})
WriteResult({ "nInserted" : 1 })
> db.categories.insert({_id:'Honorer', children:[]})
WriteResult({ "nInserted" : 1 })
> db.categories.insert({_id:'Guru', children:'['Pns','Hoorer']})
2019-03-27T21:15:59.001+0700 E QUERY    [js] SyntaxError: missing } after property list @(shell):1:46
> db.categories.insert({_id:'Guru', children:['Pns','Hoorer']})
WriteResult({ "nInserted" : 1 })
> db.categories.insert({_id:'Murid', children:[]})
WriteResult({ "nInserted" : 1 })
> db.categories.insert({_id:'Kelas', children:['Murid','Guru']})
WriteResult({ "nInserted" : 1 })
> db.categories.insert({_id:'Sekolah', children:['Kelas']})
WriteResult({ "nInserted" : 1 })
```

```
> db.categories.findOne({_id:'Guru'}).children
[ "Pns", "Hoorer" ]
```

```
> db.categories.findOne({children:'Pns'})
{ "_id" : "Guru", "children" : [ "Pns", "Hoorer" ] }
>
```

3. 3.Array Of Ancestors

C:\WINDOWS\system32\cmd.exe - mongo

```
> show databases
admin               0.000GB
childpreferences    0.000GB
config              0.000GB
hutanrimbun         0.000GB
local               0.000GB
parent_preferences  0.000GB
parentreferences    0.000GB
> use arrayancestors
switched to db arrayancestors
> db.categories.insert(_id:'Pns',ancestors:['Sekolah','Kelas','Guru'],parent:'Guru'})
2019-03-27T21:40:56.623+0700 E QUERY    [js] SyntaxError: missing ) after argument list @(shell):1:24
> db.categories.insert({_id:'Pns',ancestors:['Sekolah','Kelas','Guru'],parent:'Guru'})
WriteResult({ "nInserted" : 1 })
> db.categories.insert({_id:'Honorer',ancestors:['Sekolah','Kelas','Guru'],parent:'Guru'})
WriteResult({ "nInserted" : 1 })
> db.categories.insert({_id:'Guru',ancestors:['Sekolah','kelas'],parent:'Kelas'})
WriteResult({ "nInserted" : 1 })
> db.categories.insert({_id:'Murid',ancestors:['Sekolah','kelas'],parent:'Kelas'})
WriteResult({ "nInserted" : 1 })
> db.categories.insert({_id:'Kelas',ancestors:['Sekolah'],parent:'Sekolah'})
WriteResult({ "nInserted" : 1 })
> db.categories.insert({_id:'sekolah',ancestors:[],parent:null})
WriteResult({ "nInserted" : 1 })
```

```
> db.categories.findOne({_id:'Pns'}).ancestors
[ "Sekolah", "Kelas", "Guru" ]
```

```
> db.categories.createIndex({ancestors:1})
{
        "createdCollectionAutomatically" : false,
        "numIndexesBefore" : 1,
        "numIndexesAfter" : 2,
        "ok" : 1
}
```

```
> db.categories.find({ancestors:'Kelas'})
{ "_id" : "Pns", "ancestors" : [ "Sekolah", "Kelas", "Guru" ], "parent" : "Guru" }
{ "_id" : "Honorer", "ancestors" : [ "Sekolah", "Kelas", "Guru" ], "parent" : "Guru" }
>
```

4. Materialized Paths