Model Tree Relationships

(Project 5)

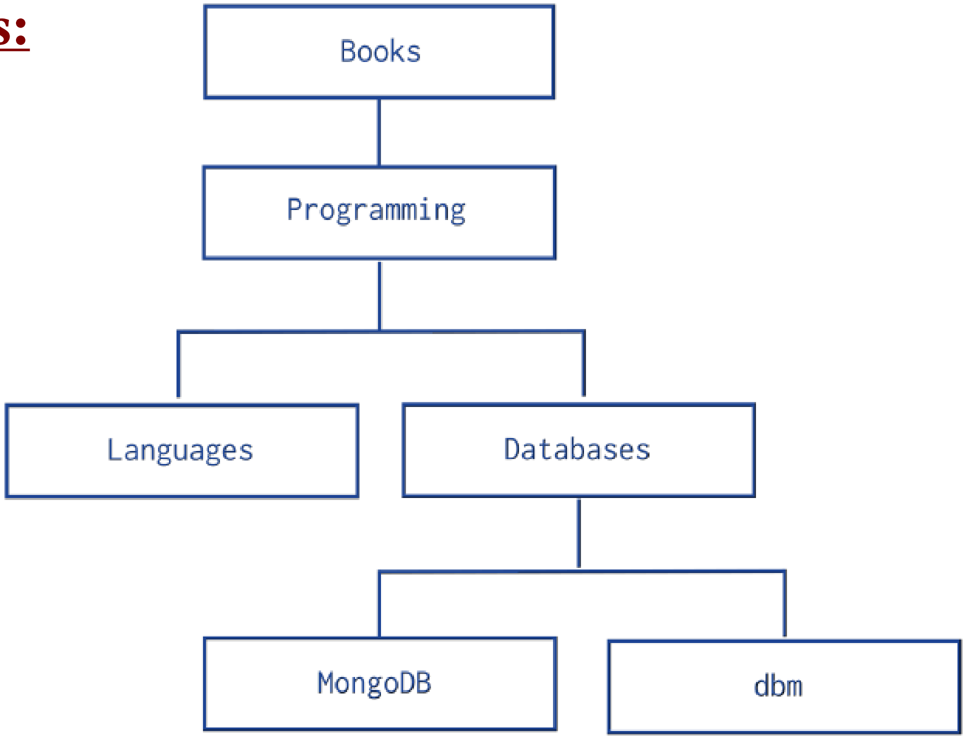
https://docs.mongodb.com/manual/applications/data-models-tree-structures/

Collections with Tree-Relationships: Modeling with References

- Records may be related to each other with these tree-like relationships

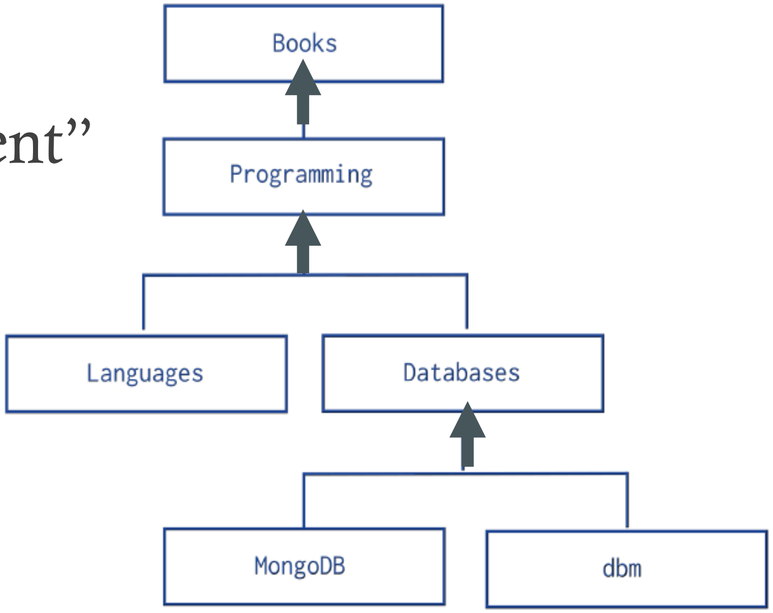
Given one node, answer queries:

- Report the parent node
- Report the children nodes
- Report the ancestors
- Report the descendants
- Report the siblings



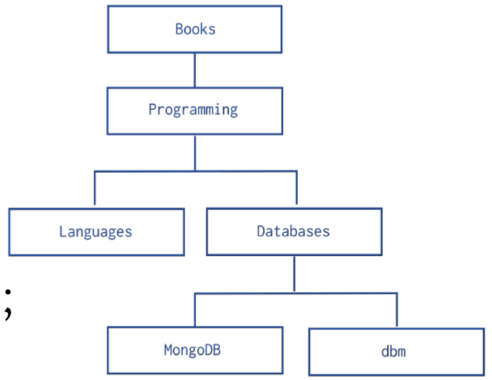
Method 1: Parent References

- Each document has a field "parent"
- Order does not matter



```
db.categories.insert( { _id: "MongoDB", parent: "Databases" } )
db.categories.insert( { _id: "dbm", parent: "Databases" } )
db.categories.insert( { _id: "Databases", parent: "Programming" } )
db.categories.insert( { _id: "Languages", parent: "Programming" } )
db.categories.insert( { _id: "Programming", parent: "Books" } )
db.categories.insert( { _id: "Books", parent: null } )
```

Method 1: Parent References



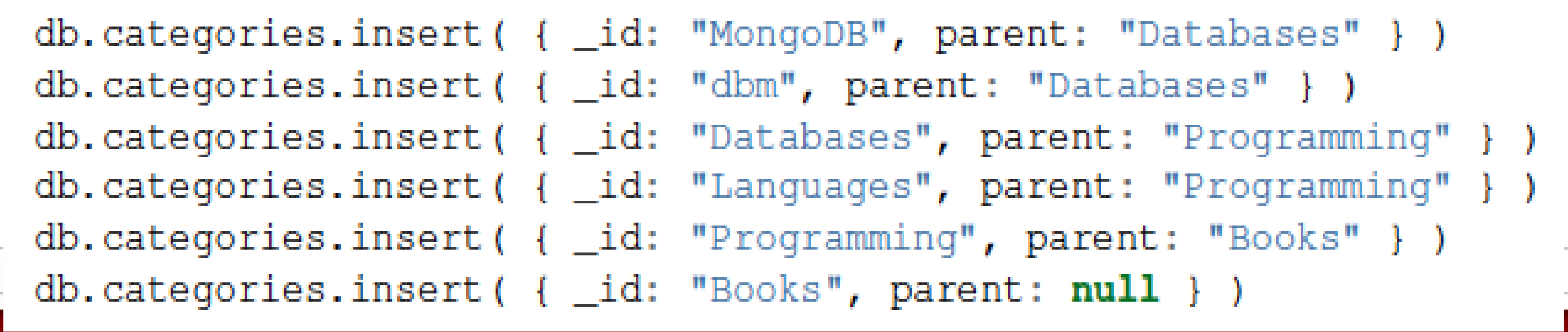
Q1: Parent of "Programming"

db.categories.findOne({ _id: " Programming " }).parent

db.categories.find({_id: "Programming"}, {parent: 1, _id: 0});

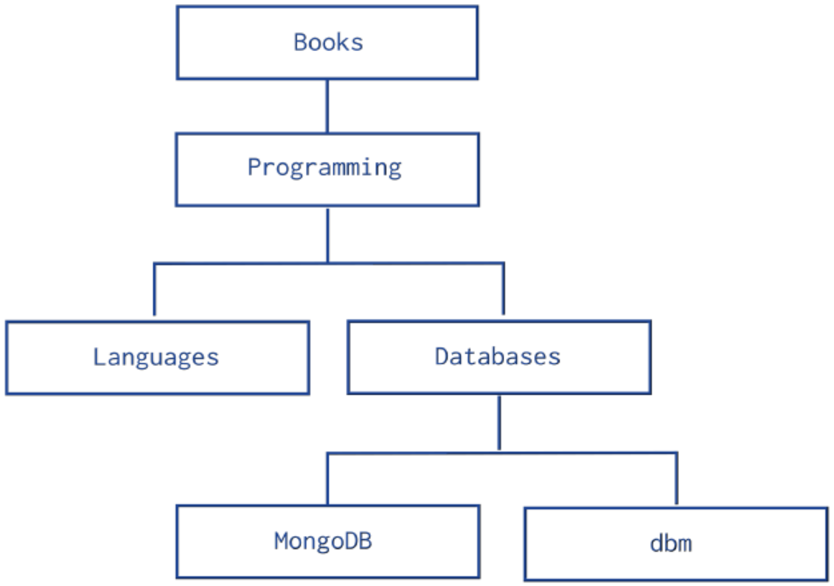
Q2: Find its immediate children node

db.categories.find({ parent: "Databases" })



```
db.categories.insert( { _id: "MongoDB", parent: "Databases" } )
db.categories.insert( { _id: "dbm", parent: "Databases" } )
db.categories.insert( { _id: "Databases", parent: "Programming" } )
db.categories.insert( { _id: "Languages", parent: "Programming" } )
db.categories.insert( { _id: "Programming", parent: "Books" } )
db.categories.insert( { _id: "Books", parent: null } )
```

Method 1: Parent References



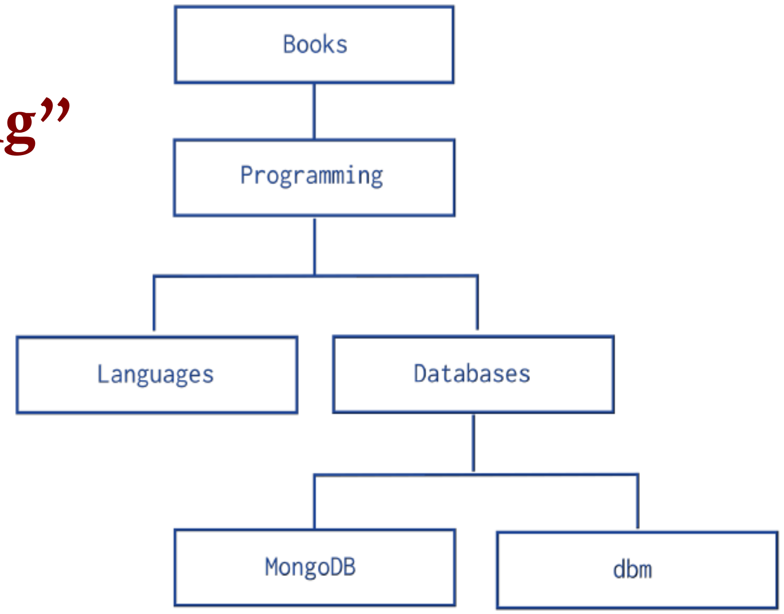
Q2: Siblings of "Databases"

```
var  parentDoc =
      db.categories.findOne( {_id: "Databases"});

  db.categories.find( {parent:  parentDoc.parent,
                                           _id: { $ne :"Databases"}      });
```

Method 1: Parent References

Q3: Descendants of "Programming"

more complex …
Requires recursive calls

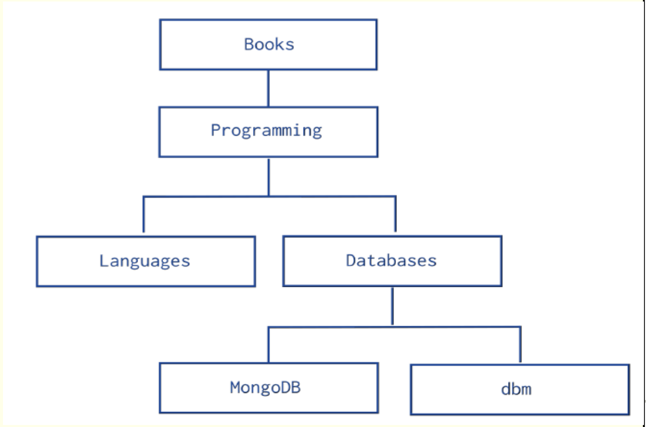


```
db.categories.insert( { _id: "MongoDB", parent: "Databases" } )
db.categories.insert( { _id: "dbm", parent: "Databases" } )
db.categories.insert( { _id: "Databases", parent: "Programming" } )
db.categories.insert( { _id: "Languages", parent: "Programming" } )
db.categories.insert( { _id: "Programming", parent: "Books" } )
db.categories.insert( { _id: "Books", parent: null } )
```

Method 1: Parent References

Q3: All descendants of "Programming"

```
var descendants = [];
var stack = [];
var item = db.categories.findOne({_id: "Programming"});
stack.push(item);
while (stack.length > 0) {
      var current = stack.pop();
       var children =  db.categories.find( {parent: current._id});
      while (children.hasNext() == true) {
            var child = children.next();
            descendants.push(child._id);
            stack.push(child);
      }
}
descendants;
```

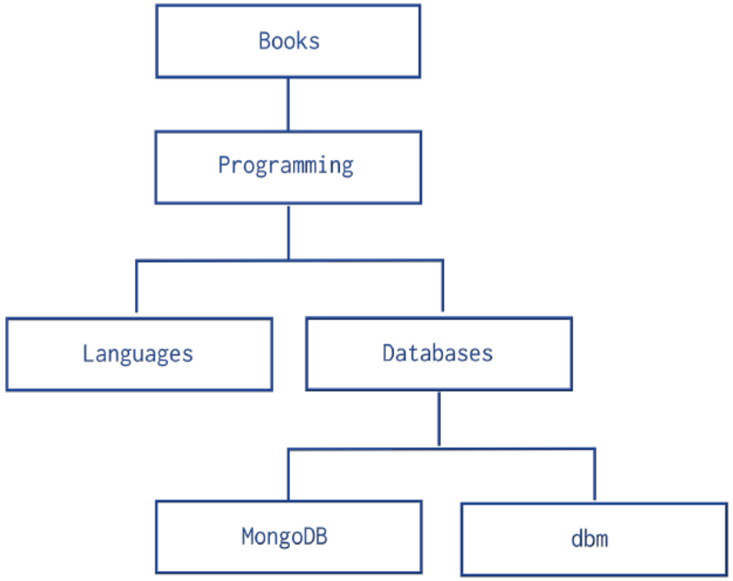


Method 1: Parent References

Q4: Ancestors of "MongoDB"

Should be:

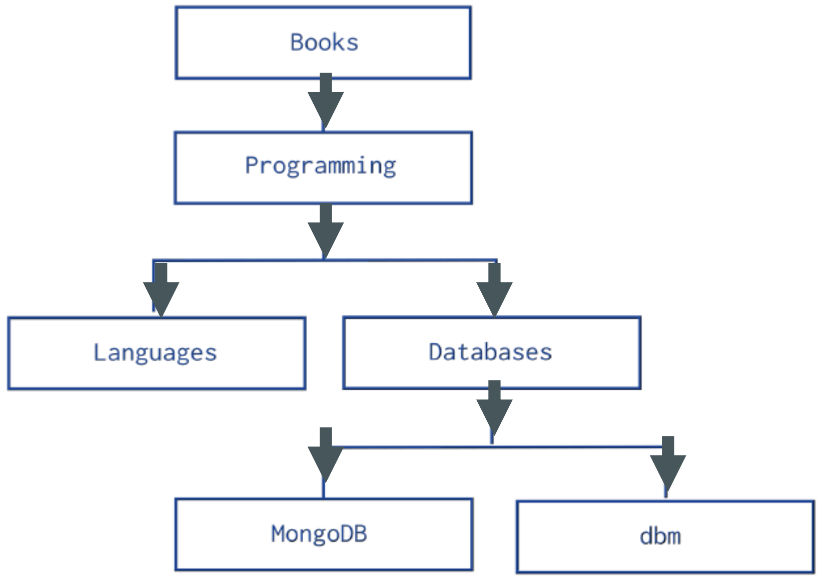
"Databases", "Programming", "Books"



```
db.categories.insert( { _id: "MongoDB", parent: "Databases" } )
db.categories.insert( { _id: "dbm", parent: "Databases" } )
db.categories.insert( { _id: "Databases", parent: "Programming" } )
db.categories.insert( { _id: "Languages", parent: "Programming" } )
db.categories.insert( { _id: "Programming", parent: "Books" } )
db.categories.insert( { _id: "Books", parent: null } )
```

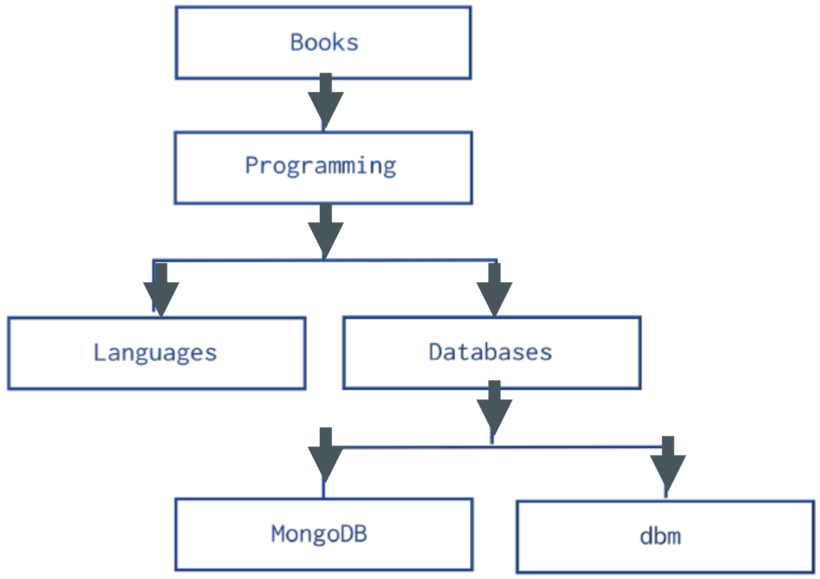
Method 2: Child References

- **How model this?**



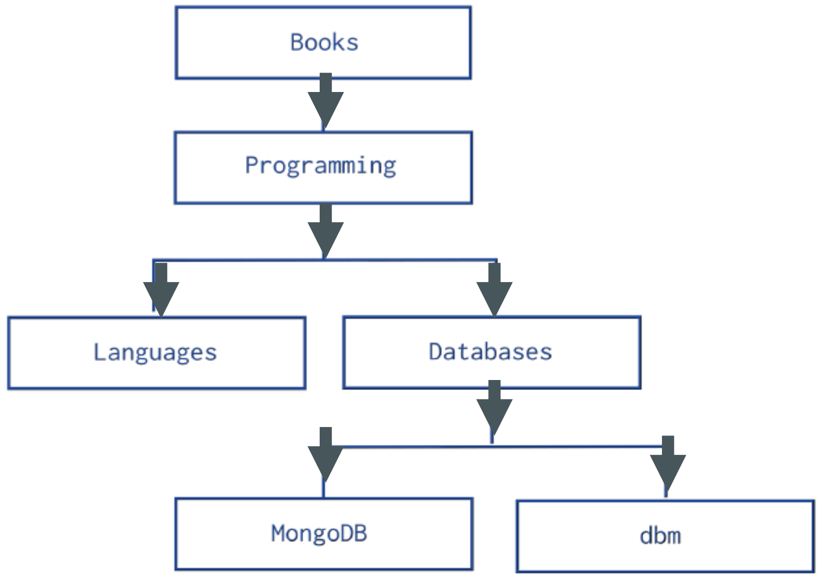
Method 2: Child References

- Each document has an array of immediate children



Method 2: Child References

- Each document has an array of immediate children



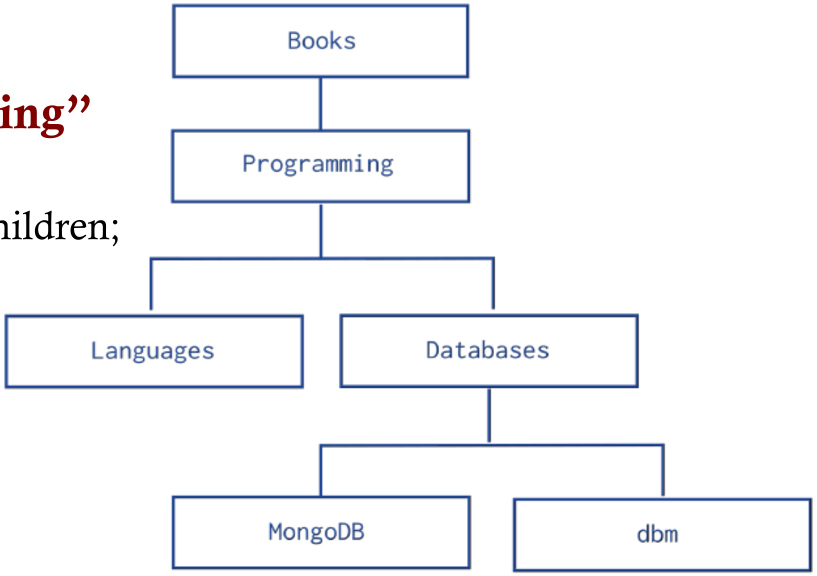
```
db.categories.insert( { _id: "MongoDB", children: [] } )
db.categories.insert( { _id: "dbm", children: [] } )
db.categories.insert( { _id: "Databases", children: [ "MongoDB", "dbm" ] } )
db.categories.insert( { _id: "Languages", children: [] } )

db.categories.insert( { _id: "Programming", children: [ "Databases", "Languages" ] } )
db.categories.insert( { _id: "Books", children: [ "Programming" ] } )
```

Method 2: Child References

Q1: Get children documents of "Programming"

var x = db.categories.findOne({_id: "Programming"}).children;

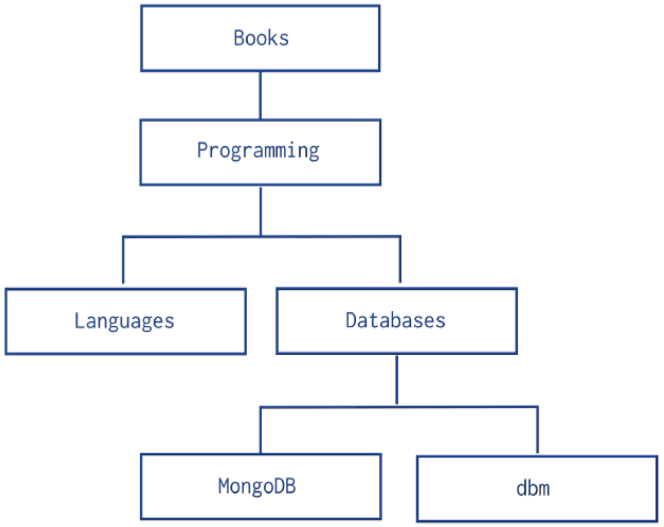
db.categories.find({_id: {$in: x}});



```
db.categories.insert( { _id: "MongoDB", children: [] } )
db.categories.insert( { _id: "dbm", children: [] } )
db.categories.insert( { _id: "Databases", children: [ "MongoDB", "dbm" ] } )
db.categories.insert( { _id: "Languages", children: [] } )

db.categories.insert( { _id: "Programming", children: [ "Databases", "Languages" ] } )
db.categories.insert( { _id: "Books", children: [ "Programming" ] } )
```

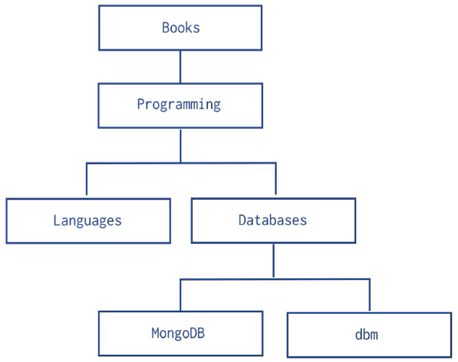
Method 2: Child References

Q2: Ancestors of “MongoDB”



```
db.categories.insert( { _id: "MongoDB", children: [] } )
db.categories.insert( { _id: "dbm", children: [] } )
db.categories.insert( { _id: "Databases", children: [ "MongoDB", "dbm" ] } )
db.categories.insert( { _id: "Languages", children: [] } )

db.categories.insert( { _id: "Programming", children: [ "Databases", "Languages" ] } )
db.categories.insert( { _id: "Books", children: [ "Programming" ] } )
```

Method 2: Child References



Q2: Ancestors of “MongoDB”

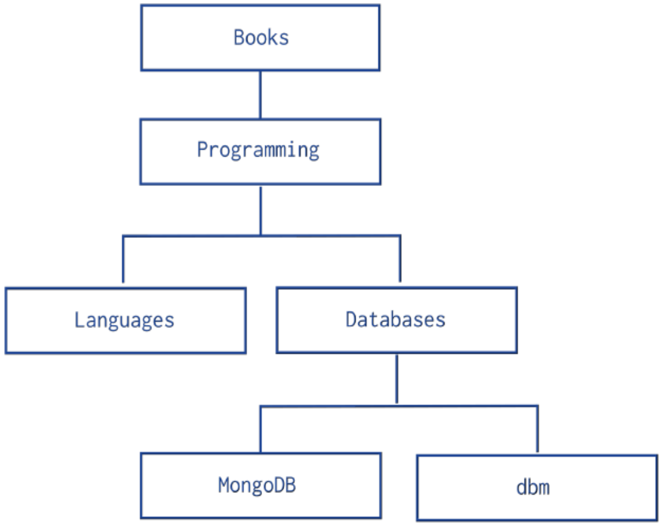
```
var results=[];
var parent = db.categories.findOne({children: "MongoDB"});
while(parent){
    print({Message: "Going up one level…"});
    results.push(parent._id);
    parent = db.categories.findOne({children: parent._id});}
results;
```

Method 2: Child References

Q3: descendants of "Books"

PROJECT 5 !!!

Should be all nodes



```
db.categories.insert( { _id: "MongoDB", children: [] } )
db.categories.insert( { _id: "dbm", children: [] } )
db.categories.insert( { _id: "Databases", children: [ "MongoDB", "dbm" ] } )
db.categories.insert( { _id: "Languages", children: [] } )

db.categories.insert( { _id: "Programming", children: [ "Databases", "Languages" ] } )
db.categories.insert( { _id: "Books", children: [ "Programming" ] } )
```

Other Methods

- **Other methods you could try :**
  - Include both parent and children
  - Include Ancestors
  - Include root-to-node path

Check MongoDB manual…

https://docs.mongodb.com/manual/applications/data-models-tree-structures/