**FINAL PROJECT**

**TEAM MEMBERS:**

* GOPICHAND BANDARUPALLI
* RAVITEJA PAGIDOJU
* SHIVA RAMA KRISHNA VODNALA

# ER DIAGRAM:

# 

show dbs

use BOOKS

show collections

newObjectId = ObjectId()

--ObjectId("5ef786b2b41463422b2e58fc")

newObjectId = ObjectId()

--ObjectId("5ef786b7b41463422b2e58fd")

newObjectId = ObjectId()

--ObjectId("5ef786beb41463422b2e58fe")

-- PUBLISHER

db.createCollection("PUBLISHER")

db.PUBLISHER.insert(

{

\_id:222,

" Name": "Sukumar Bandreddi ",

"Location": "INDIA"

})

db.PUBLISHER.insert(

{

\_id :223,

" Name": "Gopichand ",

"Location": "USA"

})

db.PUBLISHER.insert(

{

\_id :224,

" Name": "HARIKRISHNA",

"Location": "IRELAND"

})

--BOOK

db.createCollection("BOOK")

db.BOOK.insert(

{

"\_id":ObjectId("5ef786b2b41463422b2e58fc"),

" Title": "THE WINGS OF FIRE ",

YEAR: 1999,

PubID : db.PUBLISHER.find()[0].\_id})

db.BOOK.insert(

{

"\_id":ObjectId("5ef786b2b41463422b2e58fd"),

" Title": "IGNITED MINDS ",

"YEAR": "2002",

PubID :db.PUBLISHER.find()[1].\_id})

db.BOOK.insert(

{

"\_id":ObjectId("5ef786b2b41463422b2e58fe"),

" Title": " The luminous Sparks ",

"YEAR": "2004",

PubID :db.PUBLISHER.find()[2].\_id

})

--AUTHOR

db.createCollection("AUTHOR")

db.AUTHOR.insert(

{

\_id : 333,

" Name": "SAMBASIVARAO",

DOB: "14-08-1995",

})

db.AUTHOR.insert(

{

\_id :334,

" Name": "VENKY",

DOB: "14-08-1996",

})

db.AUTHOR.insert(

{

\_id : 335,

" Name": "SAMBASIVARAO",

"DOB": "14-08-1995",

})

--BOOK\_AUTHORS

db.createCollection("BOOK\_AUTHORS")

db.BOOK\_AUTHORS.insert(

{

" AuthOrder": "12345678",

BookID :db.BOOK.find()[0].\_id,

AuthID :db.AUTHOR.find()[0].\_id

})

db.BOOK\_AUTHORS.insert(

{

" AuthOrder": "12345679",

BookID:db.BOOK.find()[1].\_id,

AuthID :db.AUTHOR.find()[1].\_id

})

db.BOOK\_AUTHORS.insert(

{

" AuthOrder": "12345689",

BookID :db.BOOK.find()[2].\_id,

AuthID :db.AUTHOR.find()[2].\_id

})

--UPDATE

db.PUBLISHER.update({\_id:224},{$set:{Location: "Australia"}})

db.AUTHOR.update({\_id:334},{$set:{DOB: "14-07-1995"}})

--FIND

db.PUBLISHER.find({\_id:224}).pretty()

db.AUTHOR.find({\_id:334}).pretty()

--INDEXING

db.AUTHOR.createIndex({\_id:1})

db.PUBLISHER.ensureIndex({\_id:1})

db.BOOK.createIndex({id:1})

db.BOOK\_AUTHORS.ensureIndex({AuthOrder: 1})

Basic understanding of mongo DB with respect to RDMS.

|  |  |
| --- | --- |
| RDBMS | MongoDB |
| Database | Database |
| Table | Collection |
| Row | Document |
| Index | Index |
| Join | Embedded document, document references or lookup to combine data from different collections |

**Initial Approach:**

We tried to design our database according the ER Diagrams. Based on understanding we have on creating Tables in Relational Database, we understood the equivalents in NoSQL like Table – Collection etc. and started developing the solution. We have done some research from the internet resources (cited below) and understood more about starting the queries. As, MongoDB uses JSON documents, and stores in Binary format which is being called BSON, it’s very fast in retrieval. We were doing most of the research on linking our collections using a foreign key in Mongo DB.

In this the first command would be show db’s which will show the already created present db’s that admin,local and test in my case. Next I will use the command called use BOOKS.There is no database with the name BOOKS here but it will just use it we have never created it but in MongoDB it will never create any database or collection until we start storing documents that’s the advantage here.We can just work here and still don’t consume any memory and after exit.

Second one we created a collections Book, Publisher, Book\_Authors and Author. For example the command “db.BOOK” we use it here for collection creation.We inserted data into the collection what we created previously and command we used is “db.BOOK.insert({})”.Likewise we created different collections and inserted data into the different collections.

Thirdly we used update and Find commands.With help of these commands we can update different particular ID’S data and with the help of find commands we can find entire collection data we can observe but if we want find particular ID data that is also possible in this.

**1.** We tried to insert different formats of data as primary key i.e. int, string, Object ID.

**2.** For linking foreign key we made sure the reference is to the same collection.

We found the best part of NoSQL is that it won’t ask for schema when creating collection.

As requested in the Question we inserted 3 records into the collections.

We updated the documents according as asked in question.

**How we Chose Index:**

We selected a key for each collection to create index so that search is faster. For example we choose id PUBLISHER which is a numeric value and search is basically faster for numeric than character as each number range from 0-9 and character (0-26).So accordingly we have chosen the indexes. Also Mongo DB user B-Tree indexes which is very fast as time complexity is very less for B-Tree.

The Below table explains the comparison of RDMS action to corresponding MongoDB action.

|  |  |  |
| --- | --- | --- |
| Application | RDBMS ACTION | mongoDB Action |
| Create BookID record | INSERT to BOOK (BookID, Title,PubYear) | Insert to one document |
| Display BookID Record | SELECT and JOIN  (n) Book tables | find() single document |

**Resources:**

(w3 schools, n.d.)**(**<https://www.w3schools.com/nodejs/nodejs_mongodb_create_db.asp>**)**

(Tutorial point, n.d.)**(**<https://www.tutorialspoint.com/mongodb/mongodb_drop_database.htm>**)**

(Free online training, n.d.) **(**<https://university.mongodb.com/>**)**

(Documentation, n.d.)**(**<https://docs.mongodb.com/>)

(MongoDB Atlas database as a service for MongoDB , n.d.)**(**<https://www.mongodb.com/cloud>**)**