Nosql plan:

Links for data modelling:

<https://www.techtarget.com/searchdatamanagement/definition/data-modeling>

<https://www.erwin.com/solutions/data-modeling/data-model.aspx#:~:text=Data%20models%20are%20visual%20representations,development%20of%20effective%20information%20systems>.

<https://powerbi.microsoft.com/en-us/what-is-data-modeling/>

<https://www.gooddata.com/blog/what-a-data-model/>

links for type of schemas:

<https://www.fivetran.com/learn/what-is-a-database-schema>

[https://www.sciencedirect.com/topics/computer-science/conceptual-data-model#](https://www.sciencedirect.com/topics/computer-science/conceptual-data-model)

<https://www.gooddata.com/blog/how-build-logical-data-models-scale-analytical-applications/>

<https://www.gooddata.com/blog/physical-vs-logical-data-model/>

<https://online.visual-paradigm.com/knowledge/visual-modeling/conceptual-vs-logical-vs-physical-data-model>

does it maybe mean the shemca less, flexible schema or strict schema

links for history of RDBMS:

<https://www.linkedin.com/advice/0/what-current-trends-innovations-database#history-of-dbms>

<https://www.dataversity.net/brief-history-database-management/>

<https://www.youtube.com/watch?v=DKUJY9WgnIM>

<https://www.simplilearn.com/rise-of-nosql-and-why-it-should-matter-to-you-article>

<https://www.linkedin.com/pulse/birth-rise-nosql-pratima-upadhyay>

links for SQL vs NOSQL:

<https://www.integrate.io/blog/the-sql-vs-nosql-difference/>

Links for relationships in NoSQL:

<https://www.geeksforgeeks.org/create-relationship-in-mongodb/>

<https://www.mongodb.com/docs/manual/applications/data-models-relationships/>

<https://docs.oracle.com/cd/E17277_02/html/java/com/sleepycat/persist/model/Relationship.html>

<https://www.youtube.com/watch?v=c2M-rlkkT5o>

<https://www.mongodb.com/docs/manual/tutorial/model-embedded-one-to-one-relationships-between-documents/>

<https://docs.oracle.com/cd/E17277_02/html/java/com/sleepycat/persist/model/SecondaryKey.html>

when making your snippet code refer to this above as it shows you what to write for each relationship

notes:

you can have documents within documents hence why mongo is many to many as well.

SQL joins are harder to do so embedded documents are easier

*MONGO using shell*

Document - is a group of field value pairs to represent an object.

Collection- group of one or more documents

Database – group of one or more collection

Commands:

Show dbs. – shows all databases you have.

Use (name of one dbs.) lets you switch and use the dbs.

db.creatCollection(“”)– creates a dbs.

Eg. db.createCollection("take")

db.dropDatabase(“”) – used to drop the database ( deleted )

db.collectionname.insertOne({}) – allows you to insert values into the collection

db.collectionname.insertMany([{}, {}]) – allows you to insert more than one collections and values

eg. db.take.insertMany([{code:'A2', no: 2003, grade:68}, {code:'A3', no: 2003, grade:72}, {code:'A4', no: 2003, grade:53}, {code:'A2', no: 2005, grade:48}, {code:'A3', no: 2005, grade:52}, {code:'A2', no: 2002, grade:20}, {code:'A3', no: 2002, grade:30}, {code:'A4', no: 2002, grade:50}, {code:'B2', no: 2008, grade:90}, {code:'B2', no: 2007, grade:73}, {code:'B3', no: 2007, grade:63}])

db.collectionname.find() – finds all the relevant collections with the name

db.take.find()

https://www.youtube.com/watch?v=KA9RrZEmUNg

aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa

plan is do bullet point 4,5 and 6 and then do all the code you need and do 3 and rework 2.

or do it all in one’s so management then mechanism and then ide and conclusion?

go onto viso code and install mongo from there and the idea is to code out the tables from cw1 but using mongo and show how different it looks.

{

Links for Query Mechanism:

[Query Mechanism Tools For NoSQL - Great Learning (mygreatlearning.com)](https://www.mygreatlearning.com/no-sql/tutorials/query-mechanism-tools-for-nosql)

[Query Documents — MongoDB Manual](https://www.mongodb.com/docs/manual/tutorial/query-documents/)

CRUD with snippets should be easy just show how to use these functions in both Berkely and mongo

But do this last as it needs snippets do all your writing and research first.

<https://www.youtube.com/watch?v=DdvhZj7SsEM> - MUST WATCH

Then compare how to write mongo and Berkley against SQL using cw1.

But this time check monog 1 zip file and explain how you get it in the necessary way it is

Links for Management of NoSQL:

[What is NoSQL and how does it work  |  Google Cloud](https://cloud.google.com/discover/what-is-nosql) – not good

Maybe my management we just explain the steps of how to use the types of it. E.g. for mongo

Explain how we download it and set it up, explain how we write our code and then go on into explain the security and concurrency and etc in it.do same for Berkely and then try make 500 words on acid vs Capt.

ACID vs. CAPt – talk as much as you can about acid and do the same for cap and then compare both.

Links for ACID:

<https://www.mongodb.com/databases/acid-compliance>

<https://www.mongodb.com/basics/acid-transactions>

<https://www.databricks.com/glossary/acid-transactions#:~:text=ACID%20is%20an%20acronym%20that,operations%20are%20called%20transactional%20systems>.

<https://www.techtarget.com/searchdatamanagement/definition/ACID>

links for CAPt:

<https://www.ibm.com/topics/cap-theorem>

<https://medium.com/@kumar.barmanand/cap-theorem-and-nosql-databases-589e26e15905>

link for acid vs capt:

<https://budibase.com/blog/data/cap-vs-acid/#:~:text=Rather%2C%20CAP%20is%20a%20descriptive,it%20doesn't%20invalidate%20CAP>.

}

[How To Manage Data with MongoDB | DigitalOcean](https://www.digitalocean.com/community/tutorial-series/how-to-manage-data-with-mongodb) – this looks good as its all for mongo but does query optimisation and management so find a way in how you’re going to lay it out

<https://www.mongodb.com/docs/manual/tutorial/install-mongodb-on-windows/>

https://www.youtube.com/watch?v=HT\_TUkqK9Wg

<https://www.linkedin.com/pulse/handling-locks-mongodb-jerwin-roy#:~:text=Locking%20is%20a%20mechanism%20used,levels%20of%20locking%20in%20MongoDB>.

This above is for types of locks

Tools for Design and DEV:

[MongoDB Shell Download | MongoDB](https://www.mongodb.com/try/download/shell)

<https://restapp.io/blog/nosql-modeling-tool/>

<https://restapp.io/blog/nosql-reporting/>

both mongo are first party while Berkley is third party

but mongo can asl obe used in 3rd party

• MongoDB Compass 1.x

Conclusion:

Waffle about why NoSQL is a high rival to SQL and it depends on certain factors using examples

links in general:

<https://db-engines.com/en/system/MongoDB%3BOracle+Berkeley+DB%3BRedis>

<https://dbdb.io/db/berkeley-db>

<https://www.oracle.com/docs/tech/database/berkeley-db-xml-datasheet.pdf>

[How to Design NoSQL Databases? - RestApp](https://restapp.io/blog/how-to-design-nosql-databases/)

<https://www.digitalocean.com/community/tutorials/how-to-use-indexes-in-mongodb>

<https://docs.oracle.com/database/bdb181/html/installation/mvcc_rep_clients.html>

<https://docs.oracle.com/cd/E17277_02/html/TransactionGettingStarted/jebackuprestore.html>

useful info:

MongoDB is document store.

& Berkeley DB is a key store value, and they are both alternatives to SQL (database)

When comparing

* We first explain SQL which we have.
* Then explain NoSQL.
* So, Mongo – all its advantages and benefits
* Then Berkely
* And then compare SQL vs NoSQL / data model vs either one you choose.

ACID vs. CAPt – talk as much as you can about acid and do the same for cap and then compare both.

. History of DBMS.  
o Rise of NoSQL.

2. Data Models / Type of Schema.  
o Re-Design of existing RDBMS . . code snippets.

3. Query Mechanisms.  
o Language Support & Connectivity.  
o CRUD Functionality . . code snippets.

4. Management of NoSQL.  
o Security, Indexing, Concurrency, BackUp & Recovery.  
o ACID vs. CAPt.

5. Tools for Design & Dev.  
o IDEs . . 1st & 3rd Party.  
6. Summary & Conclusions.

|  |
| --- |
| LISTS |
| 1.Don’t ignore the similarities in core technology between Berkeley DB (a Key-Value Store) & MongoDB (a Document Store).  2. Do investigate the modelling of relationships, especially Many-to-Many, in Berkeley DB & MongoDB.  3. Do summarize the analogous concepts / functionality between Berkeley DB, MongoDB & RDBMS.  4. Do ensure your classes (i.e. Java / DPL) employ the appropriate annotations for Berkeley DB (e.g. @Entity, @PrimaryKey, @SecondaryKey).  5. Do investigate interface (i.e. Java / DPL) KeyField for composite keys in Berkeley DB.  6. Do discuss the robust capability of MongoDB’s ObjectID.  7. Don’t ignore the tabs in MongoDB Compass as they provide access to key functionality.  8. Don’t worry that some functionality is vague / missing (e.g. IDEs for Berkeley DB).  9. Do investigate support for ACID vs. CAPt (e.g. CAPt & Distributed DBs.).  10. Do ensure the report is professional looking (e.g. TofC, Glossary)  11. Don’t submit any loose code, the cwck explicitly asks for a single \*.docx file so any snippets of code must be part of your report. |

Green means confident.

Orange is maybe.

Red, is I haven’t or couldn’t be bothered.