Idea for how to do snippet: / juts copy and paste.

THIS SI NOT QUERY BUT DATA Development

I will now shortly compare the development of a DBMS with NoSQL

Starting off with SQL I will use an example of my previous project which included using SQL to create a ‘training ‘centre database which included students name, courses, grade etc.

‘

CREATE TABLE take

(code CHAR(2) ,

no INT NOT NULL,

grade TINYINT NOT NULL,

CONSTRAINT codex PRIMARY KEY (code, no),

CONSTRAINT codey FOREIGN KEY (code) REFERENCES module (code),

CONSTRAINT no1 FOREIGN KEY (no) REFERENCES delegate (no));

‘

Fig5

CREATE TABLE module

(code CHAR(2),

name VARCHAR(30) NOT NULL,

cost DECIMAL(8,2)NOT NULL,

credits TINYINT NOT NULL,

course\_code CHAR(3),

CONSTRAINT code PRIMARY KEY (code));

CREATE TABLE delegate

(no INT,

name VARCHAR(30) NOT NULL,

phone VARCHAR(30) NULL,

CONSTRAINT no PRIMARY KEY (no));

INSERT INTO take (code,no,grade)

VALUES

('A2','2003',68),

('A3','2003',72),

('A4','2003',53),

('A2','2005',48),

('A3','2005',52),

('A2','2002',20),

('A3','2002',30),

('A4','2002',50),

('B2','2008',90),

('B2','2007',73),

('B3','2007',63);

Fig6 Fig7

As you can see from the fig 5 above this is how I used SQL to create a database that contained the necessary information with the necessary results, keep in mind this is only a small section of the overall project. Fig 5 Above shows how I created a table with the create table function and the attributes that would be present in. fig6 shows the insert value which is what allowed us to tell the computer to put the values in the following order as if it was any other way there would be an error. As you will see in fig5 the table take has foreign keys this is because it is a key that is found in another table that is primary hence, making it a composite key. This was used to show the relationship between the tables. You can confirm this in fig 7 in the two tables where there is an attribute present in the tables that can be linked to the take table. Essential this is how using SQL I was able to create a database in the way I needed for my project. Now using the same project, I am going to show how it will look different in NoSQL more specifically Mongo and Berkley.

First, I will explain Mongo.

To begin I will explain first my step by step in making this. First, I used the command ‘use school’ which allowed me to use it as a database that was not yet specified but existing and after that I used the db.createCollection to create the collection called take as show in the code below.

Use school.

db. createCollection("take")

After this I was ready to get started and the first thing I had done after essentially creating the table in NoSQL was started to input the values into the collection take just to show its difference. its essentially the same with the layout but as you will see in the snippet below its more long winded as you must type what attribute it is with every new one, but it’s not that much more difficult.

{

acknowledged: true,

insertedIds: {

'0': ObjectId("65771fc41de62ba1b2accad4"),

'1': ObjectId("65771fc41de62ba1b2accad5"),

'2': ObjectId("65771fc41de62ba1b2accad6"),

'3': ObjectId("65771fc41de62ba1b2accad7"),

'4': ObjectId("65771fc41de62ba1b2accad8"),

'5': ObjectId("65771fc41de62ba1b2accad9"),

'6': ObjectId("65771fc41de62ba1b2accada"),

'7': ObjectId("65771fc41de62ba1b2accadb"),

'8': ObjectId("65771fc41de62ba1b2accadc"),

'9': ObjectId("65771fc41de62ba1b2accadd"),

'10': ObjectId("65771fc41de62ba1b2accade")

}

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},

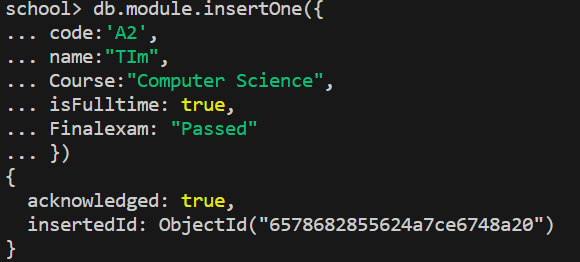
{code:'B3', no: 2007, grade:63}])

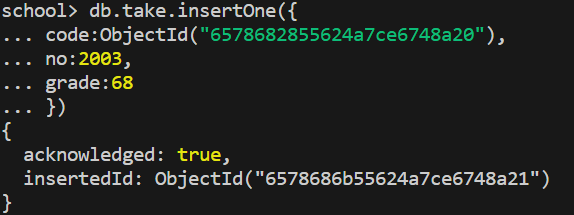
After inserting the values on the left and pressing enter we get the following message show on the right and with the function db.take.find() it shows us all of the documents in the collection

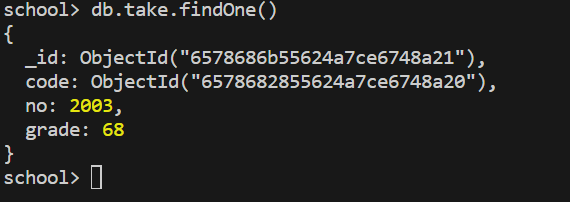
### Mongo relationships

Mongo also has its own relationships that are modelled in its own way. It doesn’t have as many as Berkely, but it does have a one-to-one and a many-to-many. Mongo can have embedded document model which basically mean that if we have two or more documents, we can embed them together to make one single document which will help the user retrieve the data using single queries rather than writing a bunch of queries.

## One-to-one

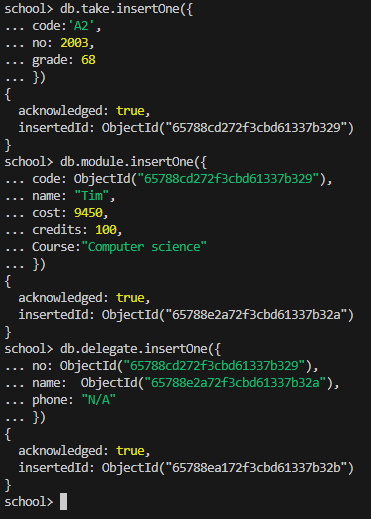
 As you can see first, we make our document that we want to show the link to another: I made up attributes of my own, but it doesn’t make a difference to the overall document. So, this is document module.



now making the second document called take you can see one difference is that the code has the object id for the table module. This is to show the relationship between the modules as you can now just look for one document that being take and if you want to find any documents that have a relationship to it you can now see the object id and find it. The snippet below shows the result of when you bring up the take documents as you can see it has its own ID and then its code links to the module document.

## Many-to-Many

There are 2 ways in which this can be showed by either making single documents multiple times, which means making three documents and having one contain all the others, or by making an embedded document which is the approach I chose to do.

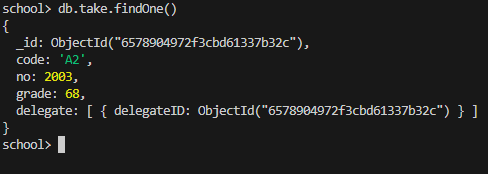


The snippet to the right is to show what I meant by making single documents to show the relationships between the documents. The main one to look at is the delegate which shows how it contains two documents linked to it showing a many to many relationships.

But as you will see this is not an embedded document as I will go on to show you in the snippet below. (Ignore the slight changes in attributes, one again it doesn’t make much of a difference)



Now this snippet is different to the previous as its to show the layout for how we would embed documents within each other. We first make the documents and once that done, we can no go and update one of the already existing documents as show in the update function. The result will be the snippet shown below.



SUMMARY >>>>?

As already the advantages/benefits of using these NoSQL have been talked about but now it comes down to which is better if any is better. SQL vs NoSQL.

There are five critical differences between SQL databases and NoSQL databases. These being SQL is relational and NoSQL is non-relational. SQL databases uses SQL and have schemas whereas NoSQL have dynamic schemas for unstructured data- meaning SQL deals with structured data and NoSQL deals with both structured and unstructured. While both are scalable SQL is vertically and NoSQL is horizontally. SQL are table based whereas NoSQL is varied on four types (refer to the four types explained earlier on).and finally SQL is better for multi-row transactions whereas NoSQL is better with unstructured data such as JSON. (Smallcombe, 2023)

NoSQL is preferred as it offers more benefits such as flexibility and scalability and the primary benefit of using this is it provides developers the ability to store and access data quickly so it’s why its typically used over SQL but just like anything else it also has its own weakness, some NoSQL can be quite resource initiative demanding high RAM AND CPU allocations (Foote, 2021), so the real question is which is better ... SQL or NoSQL?

The decision all comes down to needs and requirements of a project, for example if it’s an important project that requires fast and reliable database then it’s preferred to use a NoSQL database. But if your project is more complex than its better to use SQL.

This is how I basically got it to create the table and all I needed was the relationships. The full query on structure and referencing will come later one. https://www.youtube.com/watch?v=DdvhZj7SsEM