Databases

SQL
-) structured query language

NoSQL
⇒ not only structured query language

⇒ SQL is a database which use relational data & store it as in form of tables

⇒ SQL always comes with large no of problem like data redundancy & Irregularities.

⇒ for blank spaces in SQL is ~~NOT~~ NULL

⇒ Above problem not arised in NOSQL.

so NO SQL uses the JSON objects so no problems of NULL object arised here

⇒ mongoDB is ~~not~~ NOSQL Database so these are good & changable with ease database
⇒ mongoDB is flexible.

⇒ MongoDB has demerits as they don't have Relationships with own databases.

⇒ SQL Orders / store data of same type throught Certain kind of relation

so Database ~~decisi~~ choosing is depends on your structure

values in objects are seperated by comma

Objects in Js

```
user : {
    name : "Ghansham",
    username : "grs1712"
}
```

to store in array

```
users = [{
    name : Ghansh,
    },
    {name : "adesh",
    }
```

most Powerful SQL DB	most Powerful NoSQL DB
mySQL	mongoDB
Tables has schema can't change	Docume Struct
If want to change sth have to change all dB	Flexible to change
Relational links exist	not allot of Relations links exists

Dot operator is used to fetch data in Js objects

⇒ mongoDB arrange data in small chunks so it makes mongoDB more scalable than SQL Databases

In objects : colon is used as assignment operator

Page No.
Date

⊛ Javascript objects

objects are also variable having own desired attributes inside them

```
let person = {
        name : "Ghansham",
        edn : "Degree" ;
        hometown : "morochi";
        }
```

⇒ mostly use? let or const for declaring variable it been more precise

you can also declare like

let person = {}; or let person = new Object();

```
Person.name = "Ghanshami";
Person.hometown = "morochi";
```

Constructor to declare Person as a new object

⇒ You can't copy object but can copy object attribute

attribute = Properties

Properties are value attached with JS objects

you can use delete objectName.attribute inorder to delete a attribute

delete will remove value & attribute of object both

you can write nested objects

```
myObj = {
    name: "giri";
    age : 30,
    cars: {
        car1: "BMW",
        car2: "mercedes",
    }
}
```

↳ nested object

you can fetch data with ["i"] operator or dot operator

```
myObj.cars.car1;
myobj.cars["car1"]
myobj["cars"]["car1"]
```

three will works same

you can add function to objects

Ex

```
myobj = {
    name: function () {
        Prompt ("Enter your name");
    }
}
```

DB ⇒ Database

myobj. ~~func~~ name();

myobj.name; → It will return code for function

SQL

you can perform 4 basic operations on database with SQL

C ⇒ Create
R ⇒ Read
U ⇒ Update
D ⇒ Destroy

CRUD ⇒ 4 basic operations for an DB

create database
CREATE DATABASE name;
DROP DATABASE name;
It will delete that particular DB

① SQLiteOnline.

⇒ SQL Table

```
Syntax: CREATE TABLE tableName (
            column1 Datatype,
            column2 Datatype,
            ....
        );
```

In Database you can have multiple table & table has multiple data to store.

⇒ column1 is name of column in table
⇒ Datatype is it's datatype

Few important datatype

① char(size) :) fixed length string
size can vary betn 0 to 255 default = 1

② varchar(size) ⇒ variable length string
~~size = 255~~
generally varchar(255);

③ INT ⇒ for numerical value

~~ta~~

① Create following

Products
Id name price

⇒ keys you can assign key after declaring all column with code

PRIMARY KEY (Column name);

CREATE TABLE Products (

ID INT,
name varchar(255),
price INT,
PRIMARY KEY(ID);
);

primary key uniquely identify column

Page No.
Date

customers

ID first_name last_name address

we use command ALTER If you want change key or something else

```
APLE  ALTER TABLE Products
      ADD PRIMARY KEY(ID);
```

If you declare like below

```
ID INT NOT NULL;
```

so Id never can be blank & it is helpfull as ID is one who identifies the table element uniquely

How to insert for inserting → used when you want to fill values in specific columns

```
INSERT INTO TABLE_NAME (column1, column2 ....)
Values ( value1 ,value2 .... );
```

value1 assign for element at position column1 & so on

⇒ If your going to fill values in all column.

```
INSERT INTO TABLENAME
value ('a','b','c');
```

⇒ All value goes to 3 colum TABLENAME

insert into is command to insert data

Page No.
Date

Blankspace ⇒ NULL

⇒ If a field declared NOT NULL in schema so if you try to skip it than it will throw a error.

Finding DATA from TABLE

for it we have 3 commands

⇒ Select * from tableName;

this command will show all table data

If you want specific column

⇒ SELECT column1, column2 from TableName;

↑ It will give specific columns from table

So you want specific data with some condition on select for that we use where

⇒ SELECT column1, column2.
from tableName,
where Condition;

Ex:- Select ID
from Products
where ID = 1000;

select first_name
from Customer
where firstname
= 'Ghanshi';

Page No.
Date

⊕ Update values & Add columns in SQL Table

⊙ for update data

```
UPDATE TABLENAME
SET column1 = value, ...
where condition;
```

So update columns such only that when it needed

You can change multiple value with UPDATE table set clause call

⇒ Adding column in table

ALTER ADD command is used for that

```
ALTER TABLE Customers
ADD new_field varchar(255);
```

↑ name ↑ Datatype

You can also specify Not Null after datatype

⇒ must remember where to use Table before tableName & where to not

```
ALTER TABLE tableName;
UPDATE TableName;
```

① DELETE & DROP

DELETE From TableName
where condition;

So it will delete value corresponding to condition

whole row of table is deleted.

DELETE From Customer
Where ID = 17;

#Relationships in MySQL

Primary key (ID):
Foreign key (Customer_ID) Refers TableName
(Field which is primary key in TableName)

Ex:
create table C

id int Not NULL,
customer_id INT,
product_id INT,

Primary key (id),
Foreign (key) reference (customer, (id),

foreign key reference Product (ID);
}

Page No.
Date

which primary & foreign key we build relations

foreign key (table column) reference reference table (primary key of reference table)

It is very easy to make references from inside of table a column to target table & it's primary key

Join

inner join

It will join where foreign key match

```
SELECT ~~column~~.*
from table1
inner Join table2 on table1.col1 = table2.col2
```

Inner Join will join two tables on basis of attribute you provide after on

outer Join is also will give all other value too which not follow on condition & it is irrelevant to use it

Join will give blank if nothing is comman

⇒ inner Join will give Relational data bem multiple tables

Page No.
Date

⊛ MongoDB

CRUD Commands

mongo came with 3 predefined databases
admin
config
local

to create new db just do

use databasename → mention as you need

⇒ It will not show in show dbs where we see all databases
⇒ If a database has at least one collection can be seen in show dbs

⊛ Basic commands from mongoDB

- show db ⇒ will show all databases present in your database
- db ⇒ currently working database
- use databaseName ⇒ switch or create new database
- help ⇒ It will show helpfull element to mongoDB.

You don't have to create new ~~data base~~ database previously before using use databaseName command

#Insert data in mongoDB

So as we know mongoDB uses Json object so in order so insert we have to send Js object itself.

mongodb has two insert functions

```
db.collectionName.insertMany([
                {
                  Js object
                }
]);
```

Collection not name collectionName is created

Js object {_id:1, name: "pen"}

So

#show collections will show the database

→ for insertMany we are going to push elements in database. for that you have push a array of Javascript objects inside them

#Reading & queries in mongoDB

If you want to Read all of the value in database

db.collectionName.find();

It will give you all data inside that particular collection.

So If you want to see for ele

you have to go specific and use query & projection operators of them

In find you have to specify query in { } => Brackets

db.product.find({id: 2, "name": "Pencil"});

So It will fetch database & find someone having data like above return

Check products having values greater than something

db.product.find({price: {$gt: 1}})

↑ It will return all values having price greater than 1

↑ greater than

we can fetch all operators & make them useful

Projection
after query in find you can specify projection

So if you have 100 field & you just want name of some 2 or 3 field only so for that you can use project

select R1, R2 from table;

↑ projection

Page No.
Date

Projection is form of output you want to see after output

Ex:→

⇒ db.users.find({query}, {"id": 0, "name": 0});

here we use update func for that

⇒ db.products.updateOne({query}, {$set: {fieldwanttoadd: value}});

& it will update object

Delete inside mongoDB

Syntax:

db.collectionName.deleteOne();
db.collectionName.deleteMany();

* In paranthesis you have to pass query like

{"name": {"ghansham"}}

Even if your query provide multiple output & you are using deleteone it will delete only one

db.users.deleteMany(→ passing query as filter
{stocks: {$gte: 20}});

Relationships in mongoDB

- So you can embeded object inside another with nested objects

=> we will provide array of embeded document & that are object itself

Working with mongodb Drivers

Embed mongoDb with Nodejs

for that we will use ODM on top of mongoDb Drivers & we can say that mongoose it more good than that at native driver

Read nodejs & mongod docs