DATA SERVER
(MONGODB)

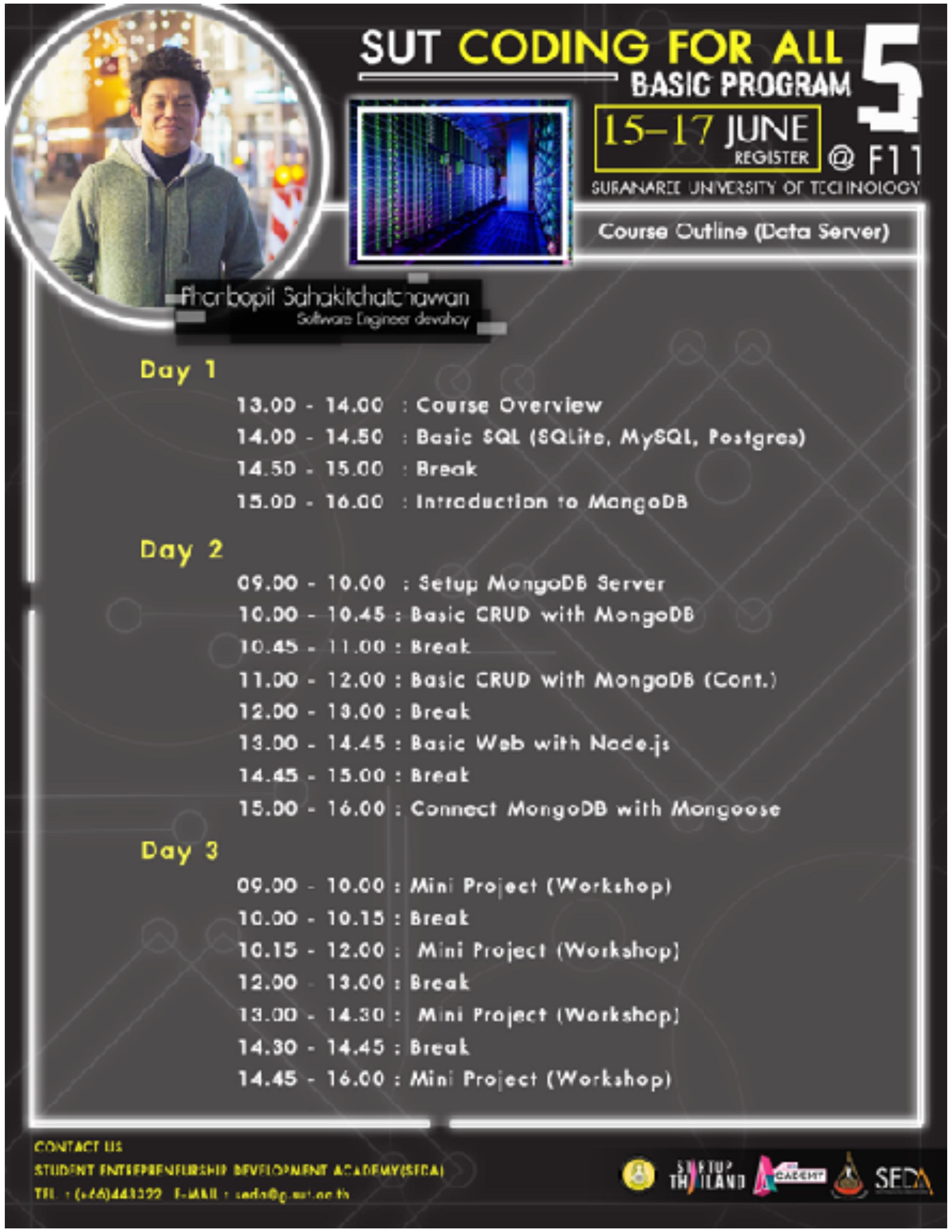
DAY 1

Phonbopit (Chai) Sahakitchatchawan

— Writer @Devahoy

Chai Phonbopit   https://devahoy.com   Phonbopit

Course Overview



SUT CODING FOR ALL 5
BASIC PROGRAM
15–17 JUNE
REGISTER
@ F11
SURANAREE UNIVERSITY OF TECHNOLOGY

Course Outline (Data Server)

Phorbopit Sahakitchatchawan
Software Engineer devahoy

Day 1

13.00 - 14.00	Course Overview
14.00 - 14.50	Basic SQL (SQLite, MySQL, Postgres)
14.50 - 15.00	Break
15.00 - 16.00	Introduction to MongoDB

Day 2

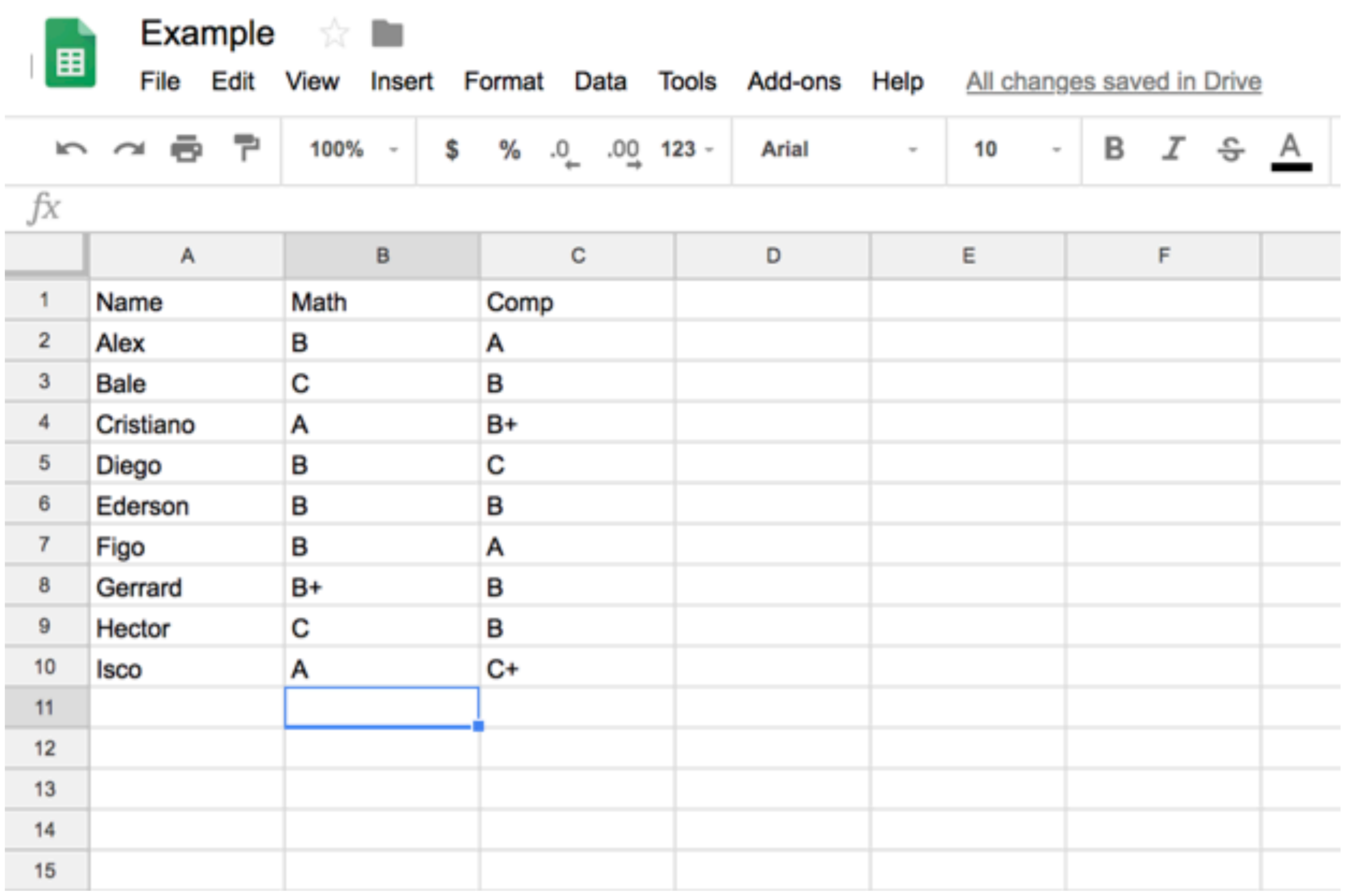
09.00 - 10.00	Setup MongoDB Server
10.00 - 10.45	Basic CRUD with MongoDB
10.45 - 11.00	Break
11.00 - 12.00	Basic CRUD with MongoDB (Cont.)
12.00 - 13.00	Break
13.00 - 14.45	Basic Web with Node.js
14.45 - 15.00	Break
15.00 - 16.00	Connect MongoDB with Mongoose

Day 3

09.00 - 10.00	Mini Project (Workshop)
10.00 - 10.15	Break
10.15 - 12.00	Mini Project (Workshop)
12.00 - 13.00	Break
13.00 - 14.30	Mini Project (Workshop)
14.30 - 14.45	Break
14.45 - 16.00	Mini Project (Workshop)

CONTACT US
STUDENT ENTREPRENEURSHIP DEVELOPMENT ACADEMY(SEDA)
TEL : (+66)443322 E-MAIL : seda@g.sut.ac.th

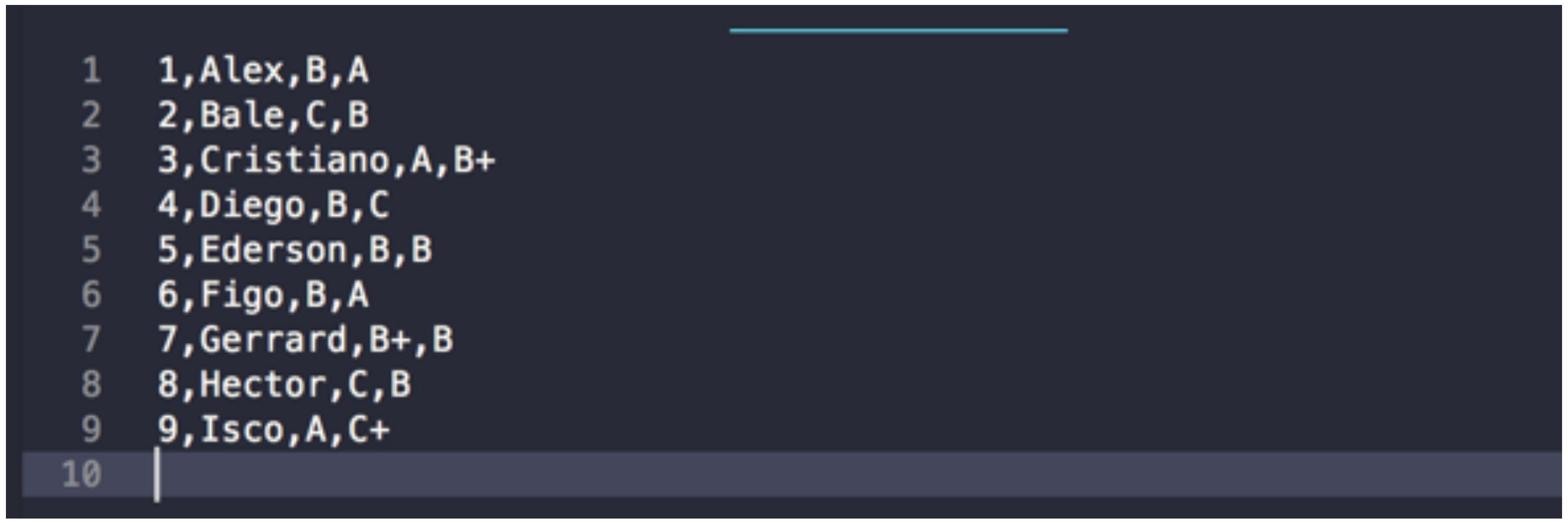
What's DATA?


Example
File Edit View Insert Format Data Tools Add-ons Help All changes saved in Drive
100% $ % .0 .00 123 Arial 10 B I S A
fx

	A	B	C	D	E	F
1	Name	Math	Comp			
2	Alex	B	A			
3	Bale	C	B			
4	Cristiano	A	B+			
5	Diego	B	C			
6	Ederson	B	B			
7	Figo	B	A			
8	Gerrard	B+	B			
9	Hector	C	B			
10	Isco	A	C+			
11						
12						
13						
14						
15						

Excel/Spreadsheet

What’s DATA?



```
1,Alex,B,A
2,Bale,C,B
3,Cristiano,A,B+
4,Diego,B,C
5,Ederson,B,B
6,Figo,B,A
7,Gerrard,B+,B
8,Hector,C,B
9,Isco,A,C+
```

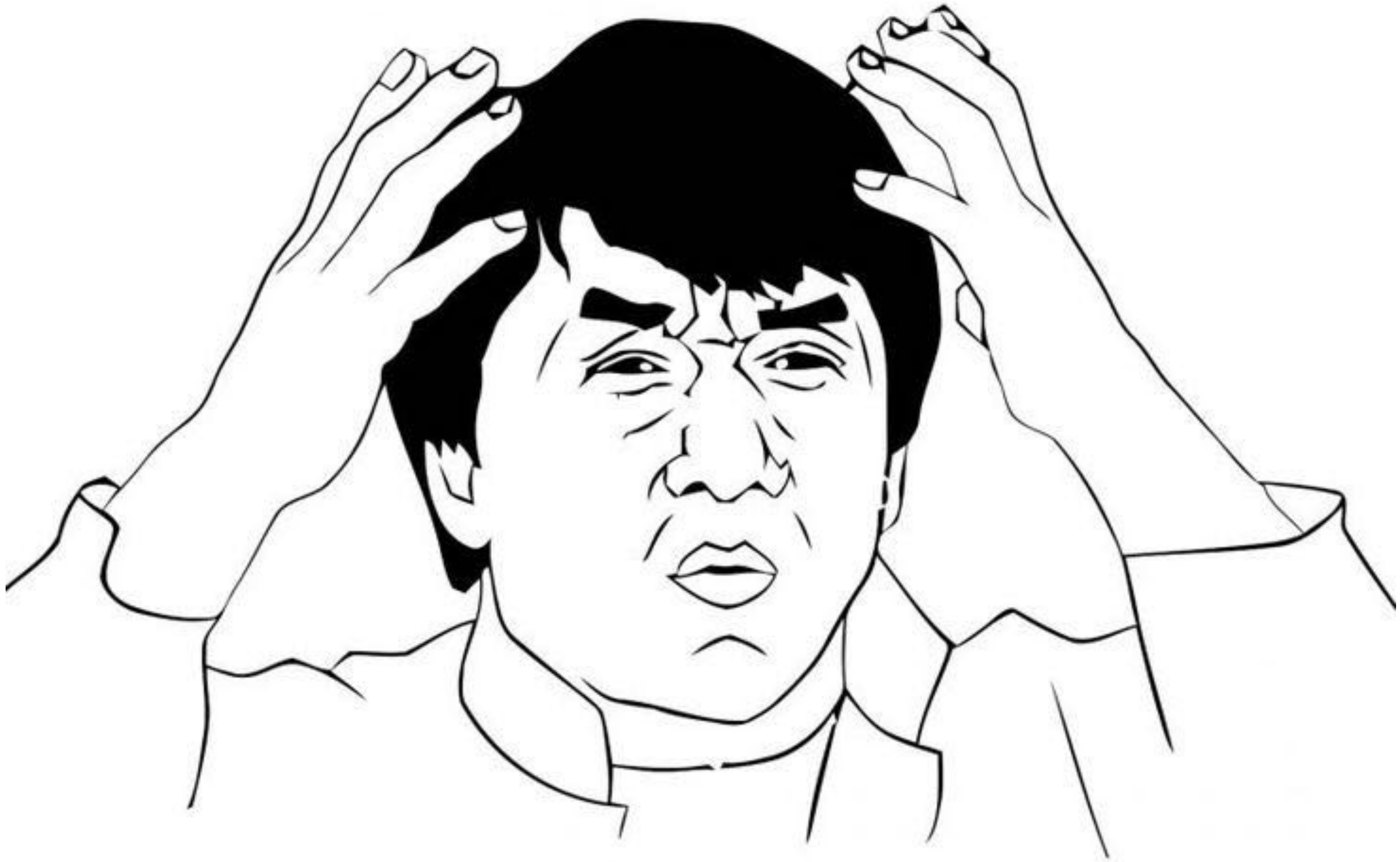
CSV

Basic SQL

- Structured Query Language
- Developed by IBM in 1970

- MySQL
- Postgres
- Microsoft SQL Server
- Oracle
- SQLite

SQL or Sequel?



Server?

Firebase

https://firebase.google.com/

Google Cloud

https://cloud.google.com/sql/

aws

https://aws.amazon.com/rds/

BASIC SQL

Create Table

```
CREATE TABLE student
(id int, name varchar(20), score int);
```

Query

```
SELECT * FROM table;

SELECT field1, field2 FROM table;
```

Query (Condition)

```sql
SELECT id, name
FROM students
WHERE score >= 80;
```

```sql
SELECT id, name
FROM students
WHERE name = 'Prayuth';
```

Query (Function)

```
SELECT avg(score)
FROM students;
```

```
SELECT count(id)
FROM students
WHERE score > 80;
```

Insert Table

```
INSERT
INTO students (id, name, score)
VALUES (2, 'Chuck Norris', 100);
```

Update* Table

```sql
UPDATE students
SET score = 99
WHERE name = 'Chuck Norris';
```

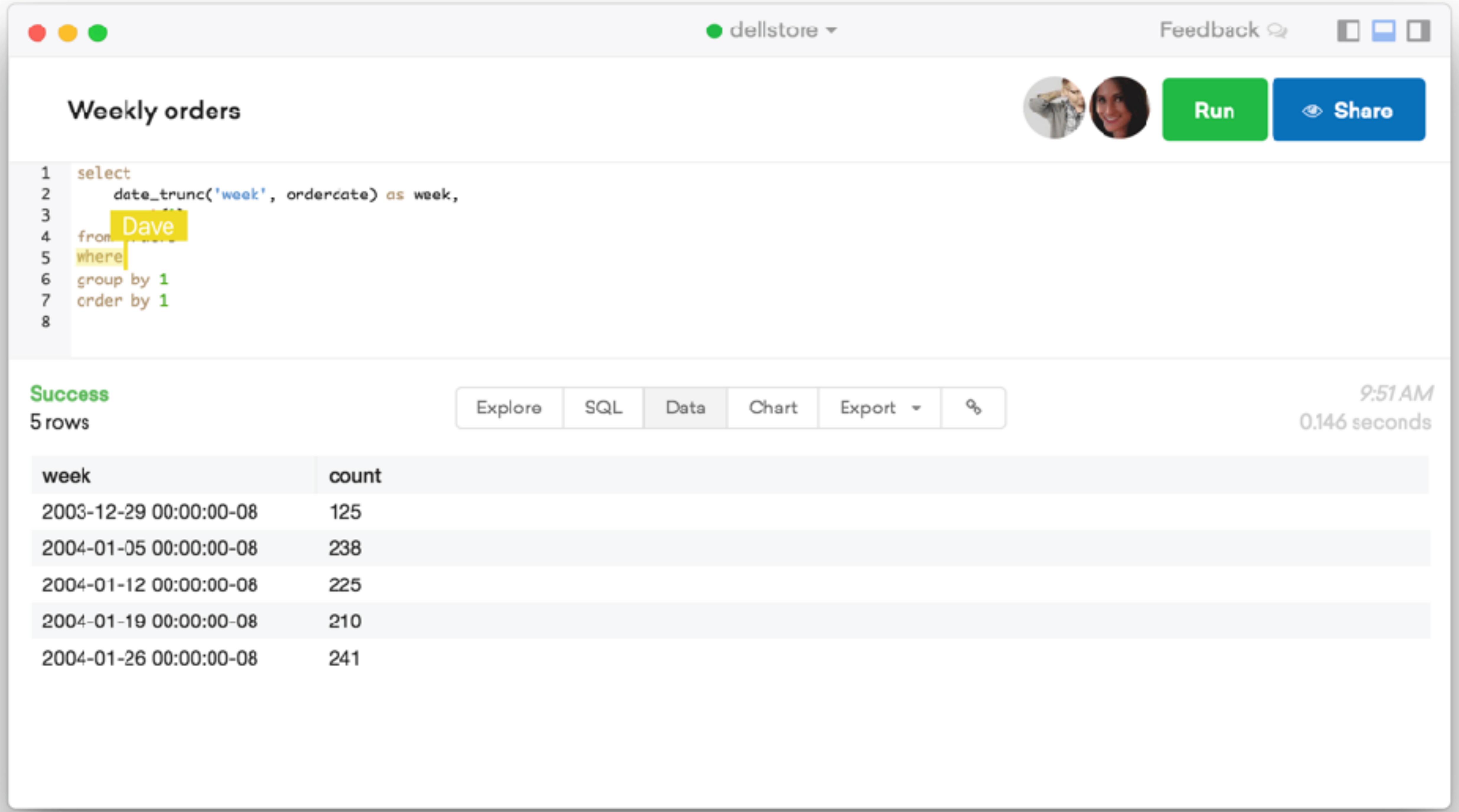
Delete* Table

```sql
DELETE FROM students
WHERE name = 'Prayuth';
```

Join Table

```
SELECT *
FROM subjects
INNER JOIN teachers
ON subjects.teacher_id = teachers.id;
```

Example



Weekly orders

```
select
    date_trunc('week', ordercate) as week,
    count(*)
from orders
where
group by 1
order by 1
```

Success
5 rows

9:51 AM
0.146 seconds

week	count
2003-12-29 00:00:00-08	125
2004-01-05 00:00:00-08	238
2004-01-12 00:00:00-08	225
2004-01-19 00:00:00-08	210
2004-01-26 00:00:00-08	241

mongoDB

What's MongoDB?

- NoSQL
- JSON-like (BSON)
- Dynamic Schema

BSON

```
{
  "_id": ObjectId("554b8ee746e04bc5503aef47"),
  "name": "Chai",
  "position": “Writer",
  "company": "Devahoy"
}
```

Dynamic Schema

```
// row 1
{
  "name": "Mark Zuckerbot"
}

// row 2
{
  "name": "Chuck Norris",
  "position": "God"
}

// row 3
{
  "name": "Chai",
  "position": "Writer",
  "company": "Devahoy"
}
```

NoSQL vs RDMS

NoSQL	RDMS
Collection	Table
Document	Row
Field	Column
Embedded, Reference	Join

Mongo Shell

```
$ mongo
MongoDB shell version v3.4.9
connecting to: mongodb://127.0.0.1:27017
MongoDB server version: 3.4.9
>
> show dbs

> use test_db
switched to db test_db
```

INSERT

```
db.collectionName.insert({})
```

https://docs.mongodb.com/manual/tutorial/insert-documents/

INSERT

```
> db.student.insert({ name: 'Chuck Norris', score: 100 })
WriteResult({ "nInserted" : 1 })

> db.student.insert({ name: 'John Doe', score: 90 })
WriteResult({ "nInserted" : 1 })
```

INSERT

```
> db.student.insert({ name: 'Chuck Norris', score: 100 })
WriteResult({ "nInserted" : 1 })

> db.student.insert({ name: 'John Doe', score: 90 })
WriteResult({ "nInserted" : 1 })
```

INSERT

```
> db.student.insert({ name: 'Chuck Norris', score: 100 })
WriteResult({ "nInserted" : 1 })

> db.student.insert({ name: 'John Doe', score: 90 })
WriteResult({ "nInserted" : 1 })
```

INSERT

```
> db.student.insert({ name: 'Chuck Norris', score: 100 })
WriteResult({ "nInserted" : 1 })

> db.student.insert({ name: 'John Doe', score: 90 })
WriteResult({ "nInserted" : 1 })
```

INSERT

```
db.student.insertOne({
  name: 'Chuck Norris'
})
```

```
db.student.insertMany([
  { name: 'Chuck Norris' },
  { name: 'John Doe', score: 90 },
  { name: 'Jane Doe', age: 25 }
])
```

QUERY

```
db.student.find();
db.student.findOne();
db.student.find(condition)
```

https://docs.mongodb.com/manual/tutorial/query-documents/index.html

QUERY

```
> db.student.findOne();
{
   "_id" : ObjectId("59a1ba277e5737fff1d675e9"),
   "name" : "Chuck Norris",
   "score" : 100
}
```

QUERY

```
> db.student.findOne();
{
   "_id" : ObjectId("59a1ba277e5737fff1d675e9"),
   "name" : "Chuck Norris",
   "score" : 100
}
```

QUERY

```
> db.student.find()

{
   "_id" : ObjectId("59a1ba277e5737fff1d675e9"),
   "name" : "Chuck Norris",
   "score" : 100
}
{
   "_id" : ObjectId("59a1ba3b7e5737fff1d675ea"),
   "name" : "John Doe",
   "score" : 90
}
```

QUERY

```
> db.student.find()

{
   "_id" : ObjectId("59a1ba277e5737fff1d675e9"),
   "name" : "Chuck Norris",
   "score" : 100
}
{
   "_id" : ObjectId("59a1ba3b7e5737fff1d675ea"),
   "name" : "John Doe",
   "score" : 90
}
```

Query Operator

- $and, $or
- $lt, $lte, $gt, $gte
- $in, $nin, $eq, $ne

https://docs.mongodb.com/manual/reference/operator/query/#query-selectors

Query Operator

```
> db.student.find( { score: { $gt: 95 } } )

{
  "_id" : ObjectId("59a1ba277e5737fff1d675e9"),
  "name" : "Chuck Norris",
  "score" : 100
}
```

Query Operator

```
> db.student.find( { score: { $gt: 95 } } )

{
   "_id" : ObjectId("59a1ba277e5737fff1d675e9"),
   "name" : "Chuck Norris",
   "score" : 100
}
```

UPDATE

```
db.collection.updateOne(<filter>, <update>)
```

https://docs.mongodb.com/manual/tutorial/update-documents/

UPDATE

```
> db.student.updateOne( { score: 90 }, { $set: { score: 95 } } )
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }

> db.student.find().pretty()
{
  "_id" : ObjectId("59a1ba277e5737fff1d675e9"),
  "name" : "Chuck Norris",
  "score" : 100
}
{
  "_id" : ObjectId("59a1ba3b7e5737fff1d675ea"),
  "name" : "John Doe",
  "score" : 95
}
```

UPDATE

```
> db.student.updateOne( { score: 90 }, { $set: { score: 95 } } )
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }

> db.student.find().pretty()
{
  "_id" : ObjectId("59a1ba277e5737fff1d675e9"),
  "name" : "Chuck Norris",
  "score" : 100
}
{
  "_id" : ObjectId("59a1ba3b7e5737fff1d675ea"),
  "name" : "John Doe",
  "score" : 95
}
```

UPDATE

```
> db.student.updateOne( { score: 90 }, { $set: { score: 95 } } )
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }

> db.student.find().pretty()
{
  "_id" : ObjectId("59a1ba277e5737fff1d675e9"),
  "name" : "Chuck Norris",
  "score" : 100
}
{
  "_id" : ObjectId("59a1ba3b7e5737fff1d675ea"),
  "name" : "John Doe",
  "score" : 95
}
```

UPDATE

```
> db.student.updateOne( { score: 90 }, { $set: { score: 95 } } )
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }

> db.student.find().pretty()
{
  "_id" : ObjectId("59a1ba277e5737fff1d675e9"),
  "name" : "Chuck Norris",
  "score" : 100
}
{
  "_id" : ObjectId("59a1ba3b7e5737fff1d675ea"),
  "name" : "John Doe",
  "score" : 95
}
```

UPDATE

```
> db.student.updateOne( { score: 90 }, { $set: { score: 95 } } )
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }

> db.student.find().pretty()
{
  "_id" : ObjectId("59a1ba277e5737fff1d675e9"),
  "name" : "Chuck Norris",
  "score" : 100
}
{
  "_id" : ObjectId("59a1ba3b7e5737fff1d675ea"),
  "name" : "John Doe",
  "score" : 95
}
```

UPDATE

```
> db.student.updateOne( { score: 90 }, { $set: { score: 95 } } )
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }

> db.student.find().pretty()
{
  "_id" : ObjectId("59a1ba277e5737fff1d675e9"),
  "name" : "Chuck Norris",
  "score" : 100
}
{
  "_id" : ObjectId("59a1ba3b7e5737fff1d675ea"),
  "name" : "John Doe",
  "score" : 95
}
```

DELETE

```
db.collection.deleteOne()
```

https://docs.mongodb.com/manual/tutorial/remove-documents/

DELETE

```
> db.student.deleteOne( {score: 95} )
{ "acknowledged" : true, "deletedCount" : 1 }

> db.student.find().pretty()
{
  "_id" : ObjectId("59a1ba277e5737fff1d675e9"),
  "name" : "Chuck Norris",
  "score" : 100
}
```

DELETE

```
> db.student.deleteOne( {score: 95} )
{ "acknowledged" : true, "deletedCount" : 1 }

> db.student.find().pretty()
{
  "_id" : ObjectId("59a1ba277e5737fff1d675e9"),
  "name" : "Chuck Norris",
  "score" : 100
}
```

DELETE

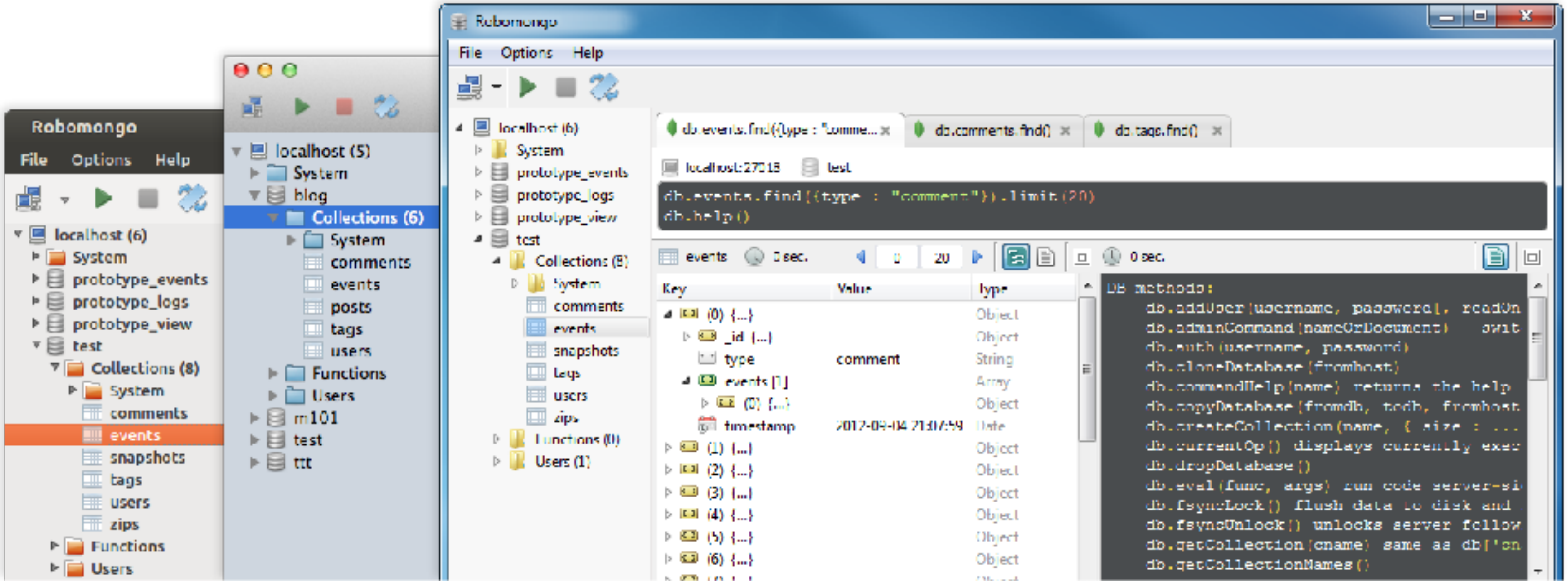
```
> db.student.deleteOne( {score: 95} )
{ "acknowledged" : true, "deletedCount" : 1 }

> db.student.find().pretty()
{
  "_id" : ObjectId("59a1ba277e5737fff1d675e9"),
  "name" : "Chuck Norris",
  "score" : 100
}
```

DELETE

```
> db.student.deleteOne( {score: 95} )
{ "acknowledged" : true, "deletedCount" : 1 }

> db.student.find().pretty()
{
  "_id" : ObjectId("59a1ba277e5737fff1d675e9"),
  "name" : "Chuck Norris",
  "score" : 100
}
```

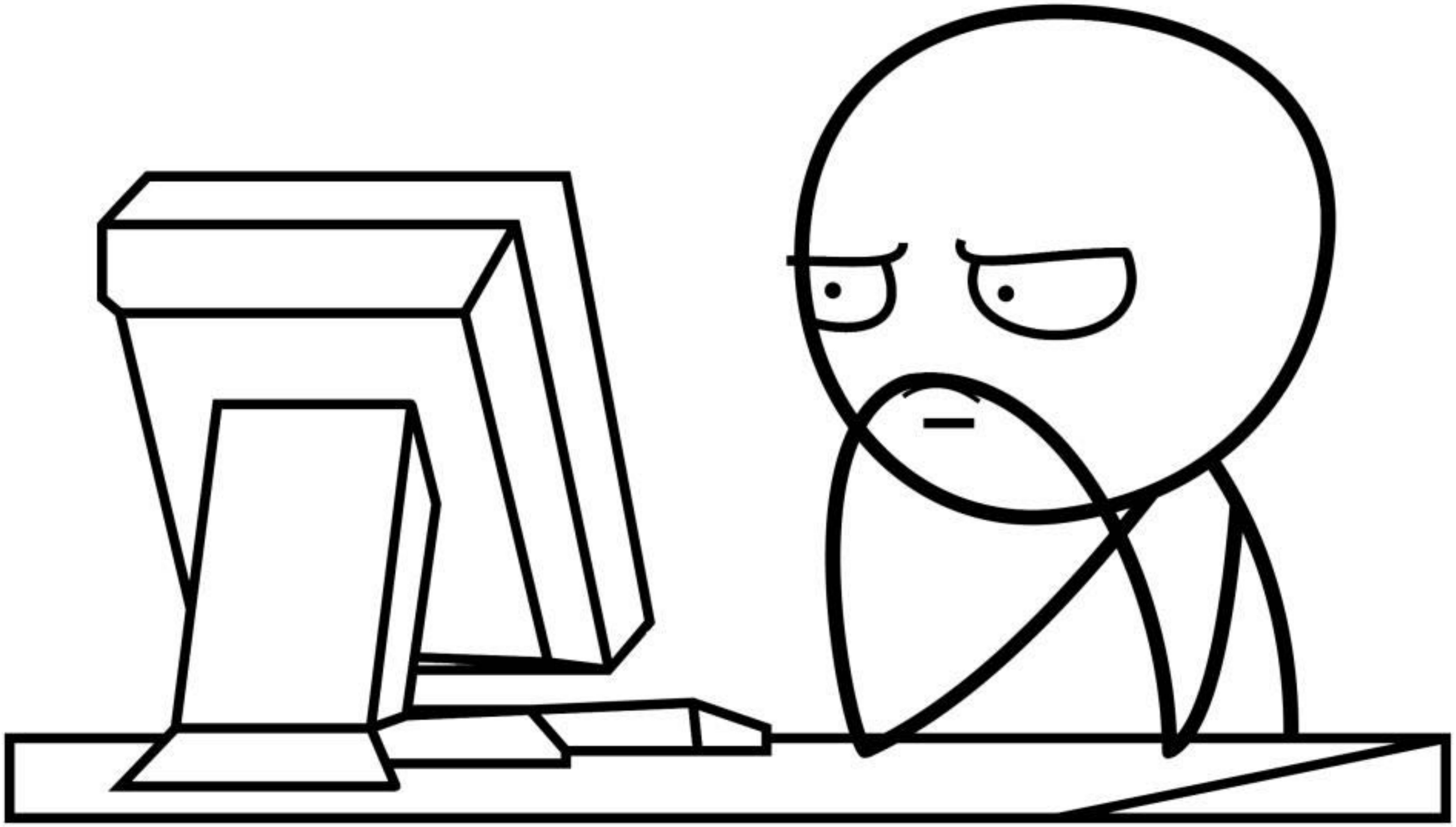


```
db.events.find({type : "comment"}).limit(20)
db.help()
```

```
DB methods:
    db.addUser(username, password[, readOn
    db.adminCommand(nameOrDocument)   swit
    db.auth(username, password)
    db.cloneDatabase(fromhost)
    db.commandHelp(name) returns the help
    db.copyDatabase(fromdb, todb, fromhost
    db.createCollection(name, { size : ...
    db.currentOp() displays currently exec
    db.dropDatabase()
    db.eval(func, args) run code server-si
    db.fsyncLock() flush data to disk and
    db.fsyncUnlock() unlocks server follow
    db.getCollection(cname) same as db['cn
    db.getCollectionNames()
```

Key	Value	Type
(0) {...}		Object
_id {...}		Object
type	comment	String
events [1]		Array
(0) {...}		Object
timestamp	2012-09-04 21:07:59	Date
(1) {...}		Object
(2) {...}		Object
(3) {...}		Object
(4) {...}		Object
(5) {...}		Object
(6) {...}		Object

https://robomongo.org/



https://bit.ly/sut-ds-starter