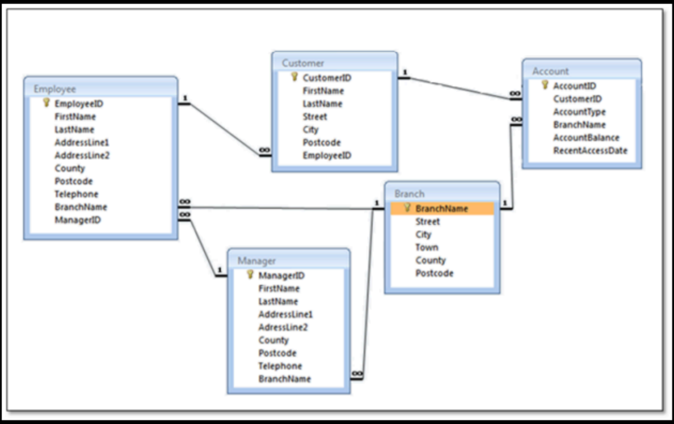
SQL Databases



Structured Query Language

Relational Databases

→structure for tables

→Querying is based on tables to have some column in common

SQL Databases

Structured Query Language

Relational Databases

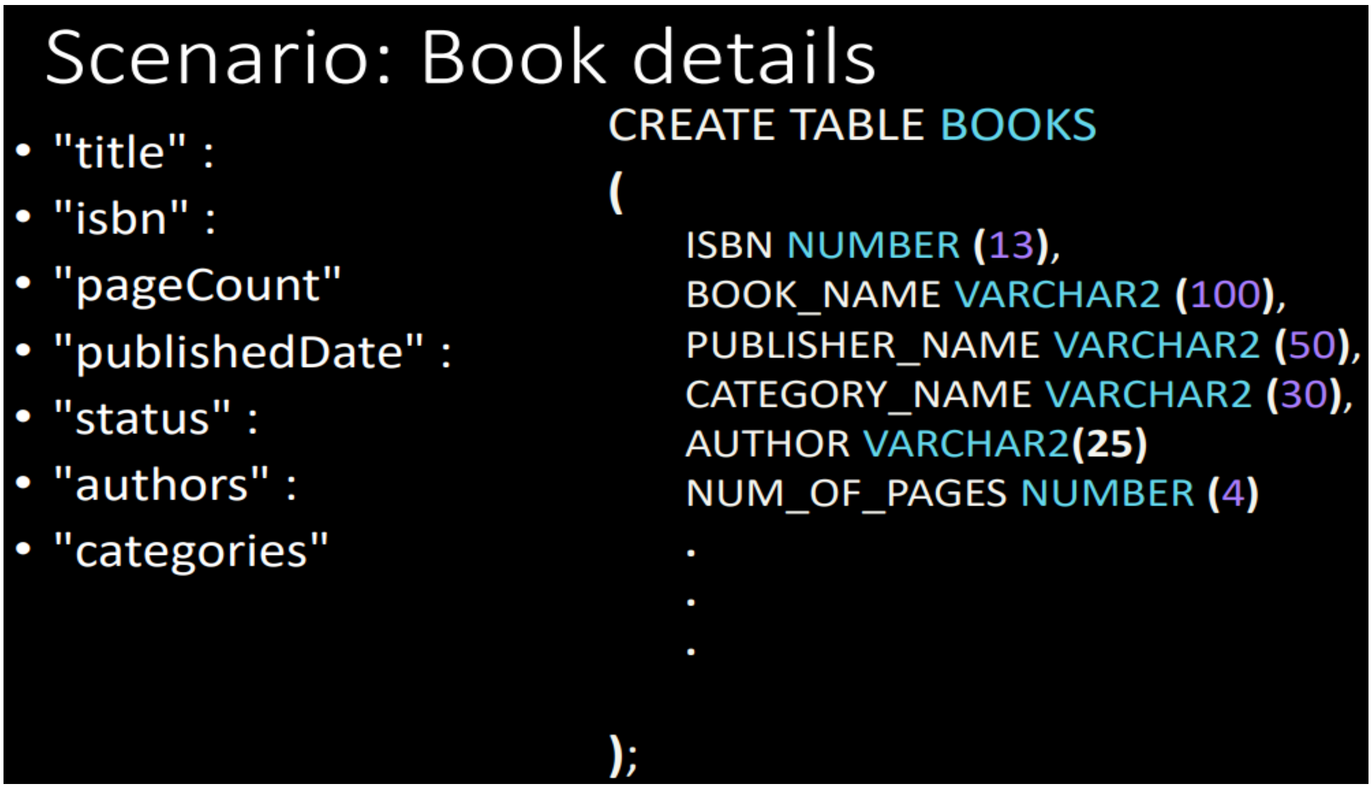
→structure for tables

→Querying is based on tables to have some column in common

→Having, group by etc

→Join

→Aggregation

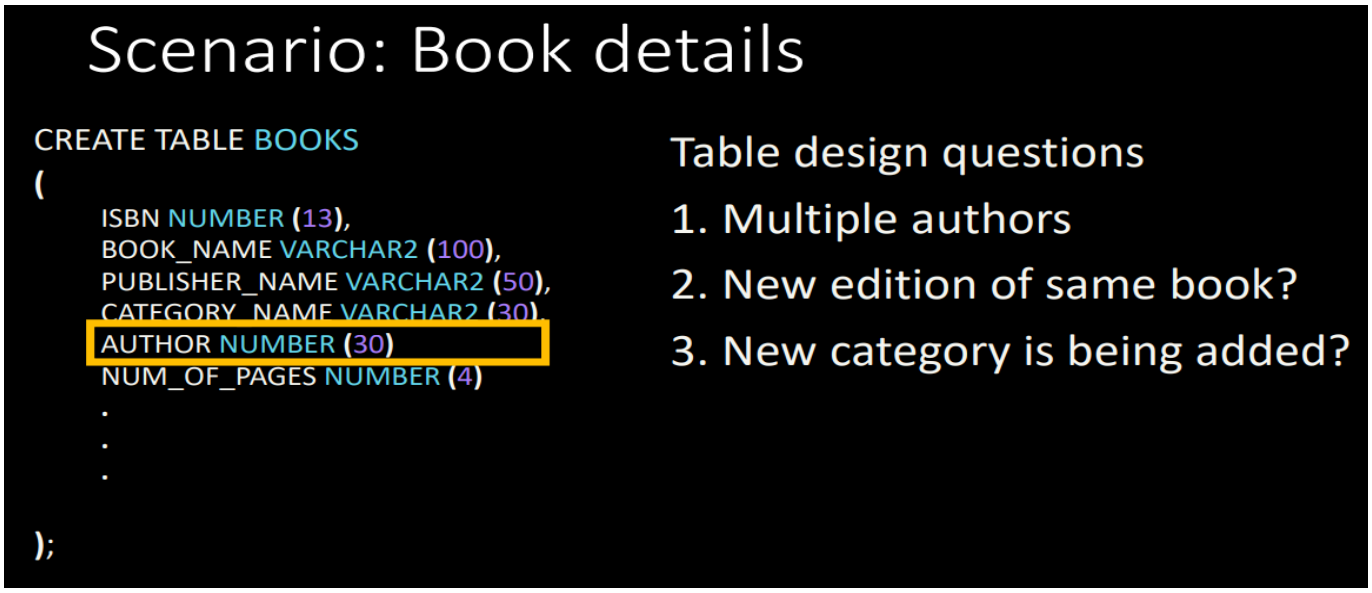


Scenario: Book details

- "title" :
- "isbn" :
- "pageCount"
- "publishedDate" :
- "status" :
- "authors" :
- "categories"

```
CREATE TABLE BOOKS
(
    ISBN NUMBER (13),
    BOOK_NAME VARCHAR2 (100),
    PUBLISHER_NAME VARCHAR2 (50),
    CATEGORY_NAME VARCHAR2 (30),
    AUTHOR VARCHAR2(25)
    NUM_OF_PAGES NUMBER (4)
    .
    .
    .

);
```



Scenario: Book details

```
CREATE TABLE BOOKS
(
    ISBN NUMBER (13),
    BOOK_NAME VARCHAR2 (100),
    PUBLISHER_NAME VARCHAR2 (50),
    CATEGORY_NAME VARCHAR2 (30),
    AUTHOR NUMBER (30)
    NUM_OF_PAGES NUMBER (4)
    .
    .
    .

);
```

Table design questions

1. Multiple authors
2. New edition of same book?
3. New category is being added?

SQL Structure

-
```
CREATE TABLE table_name
(
    column1 datatype,
    column2 datatype,
    column3 datatype,
   ....
);
```

- Structured
- Predefined schema

➢Fixed rows and columns
➢Changes to tables is not possible
➢Agile environment

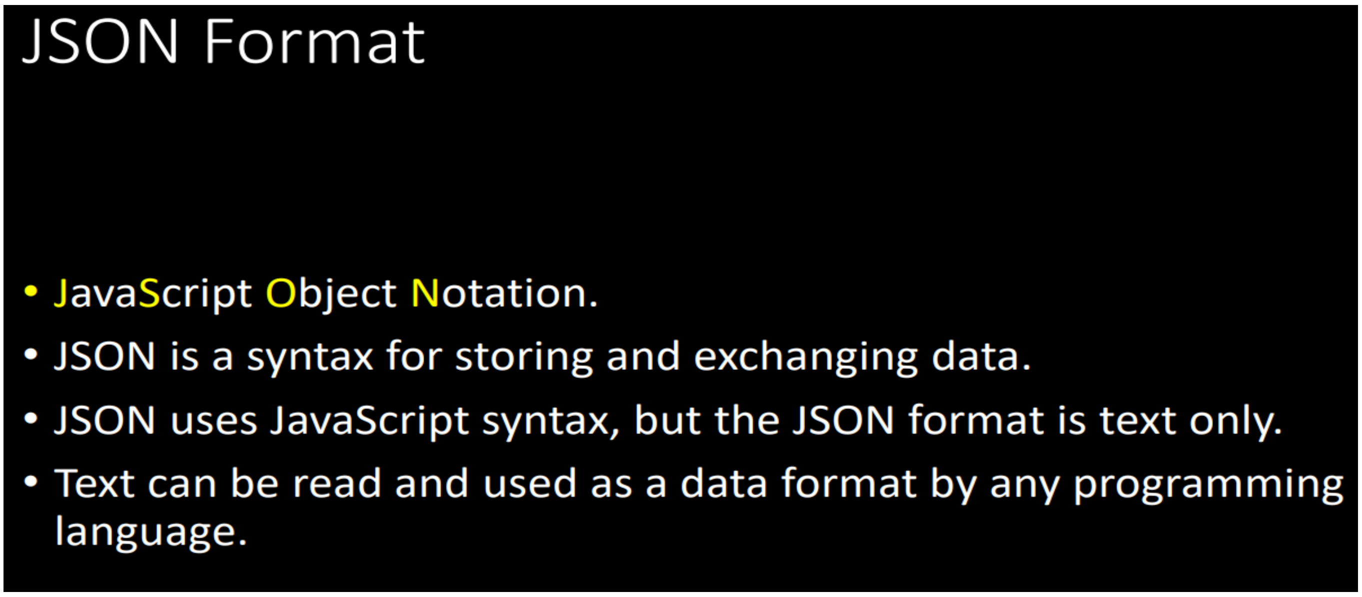
Online data

- Unstructured
- Text
- Platform independent
- Browser supported data

- Twitter data
- Facebook data
- Stock Exchange data
- JSON Data Format

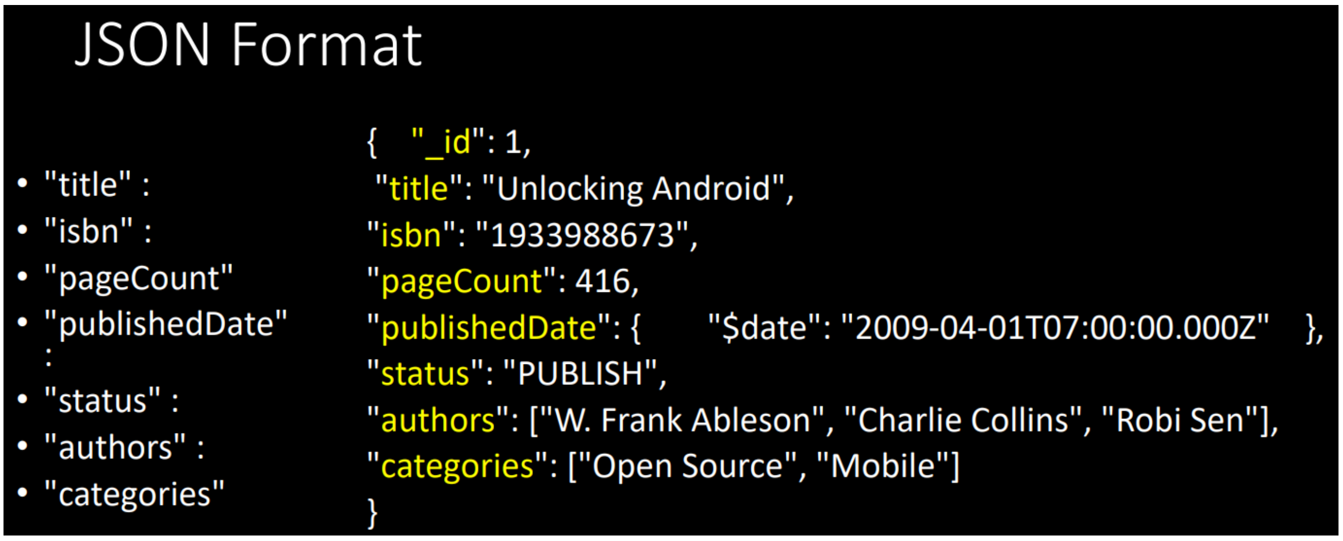
JSON Format

- JavaScript Object Notation.



JSON Format

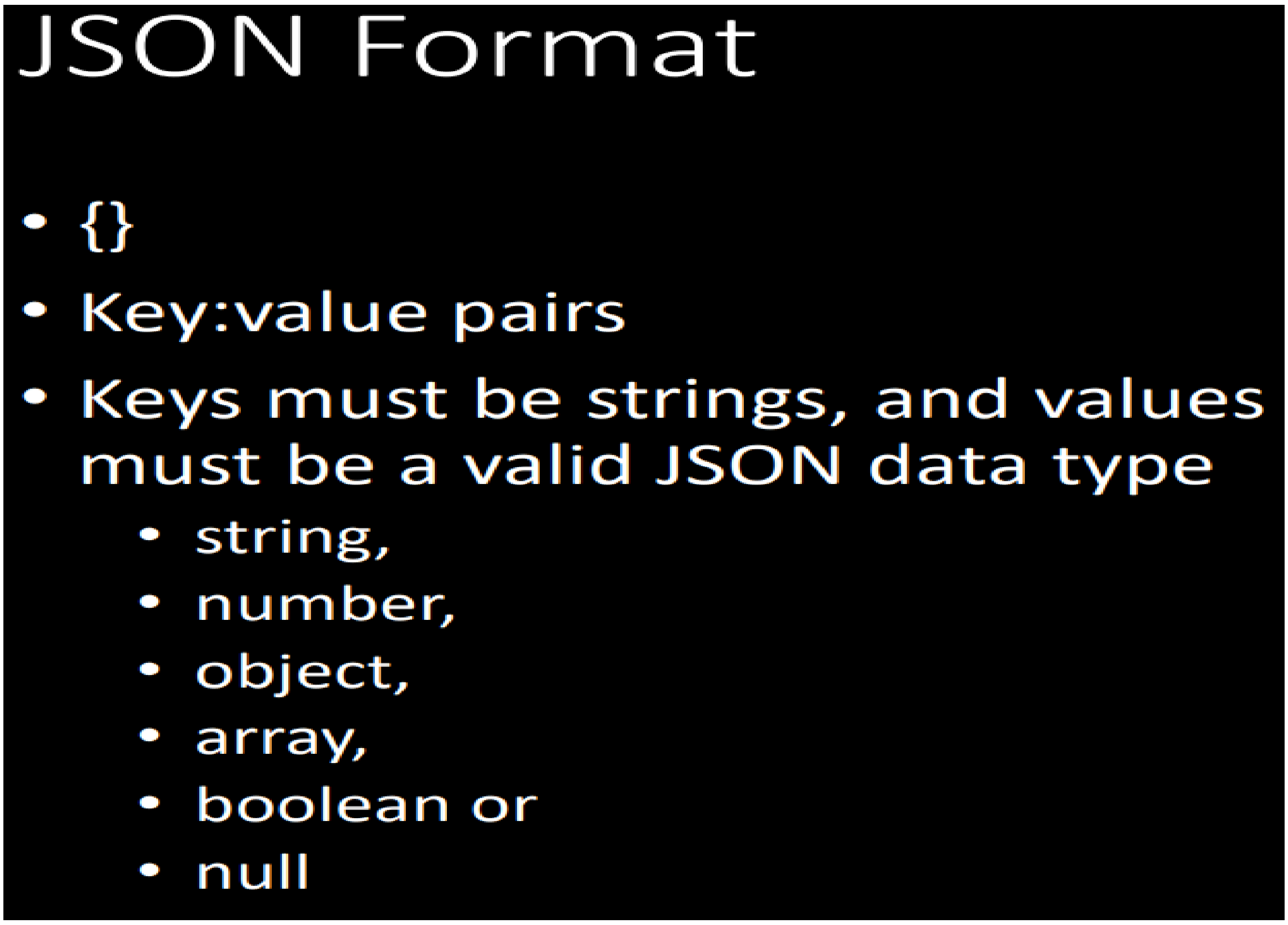
- JavaScript Object Notation.
- JSON is a syntax for storing and exchanging data.
- JSON uses JavaScript syntax, but the JSON format is text only.
- Text can be read and used as a data format by any programming language.



JSON Format

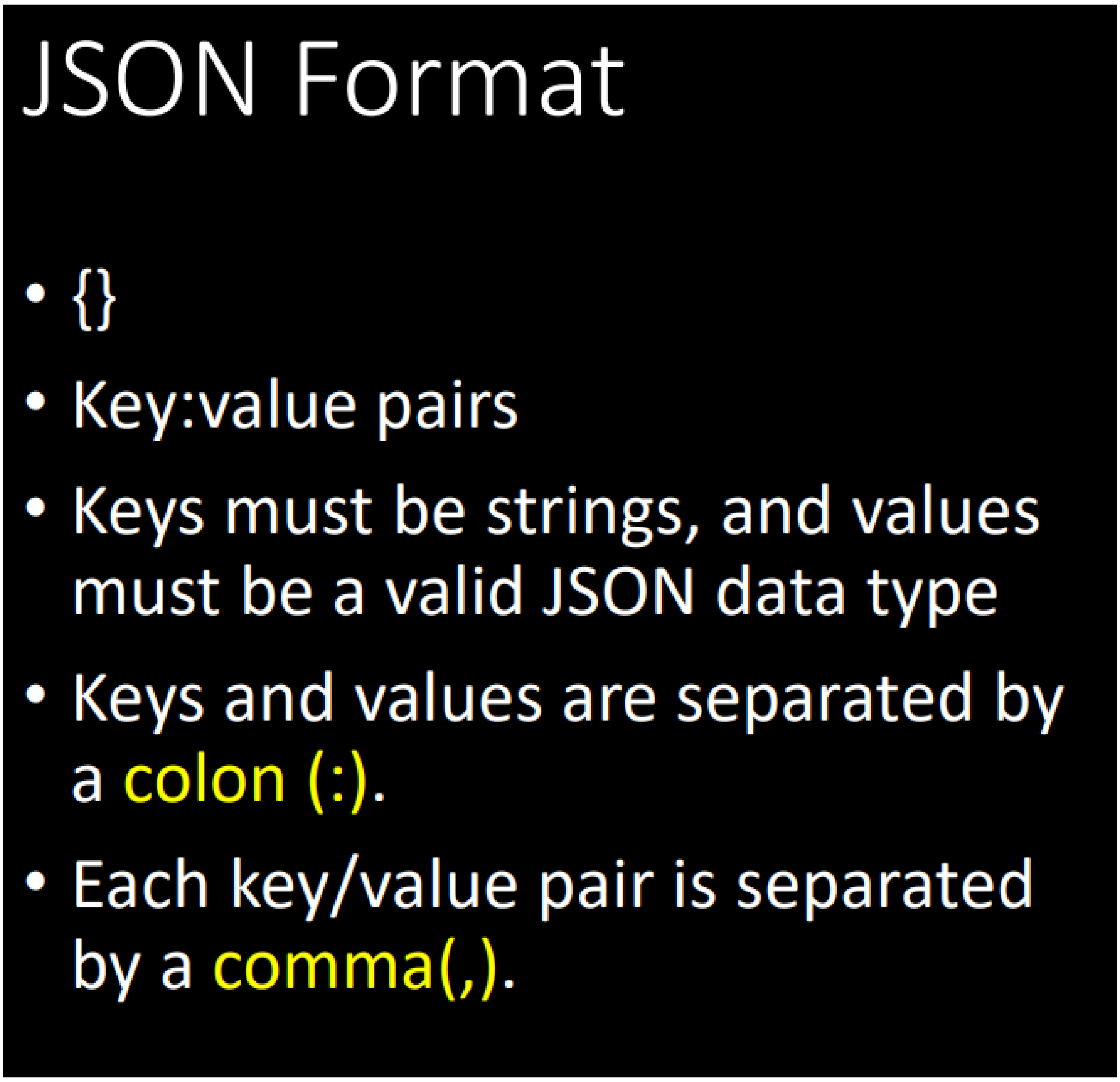
- "title" :
- "isbn" :
- "pageCount"
- "publishedDate" :
- "status" :
- "authors" :
- "categories"

```
{    "_id": 1,
 "title": "Unlocking Android",
"isbn": "1933988673",
"pageCount": 416,
"publishedDate": {       "$date": "2009-04-01T07:00:00.000Z"    },
"status": "PUBLISH",
"authors": ["W. Frank Ableson", "Charlie Collins", "Robi Sen"],
"categories": ["Open Source", "Mobile"]
}
```



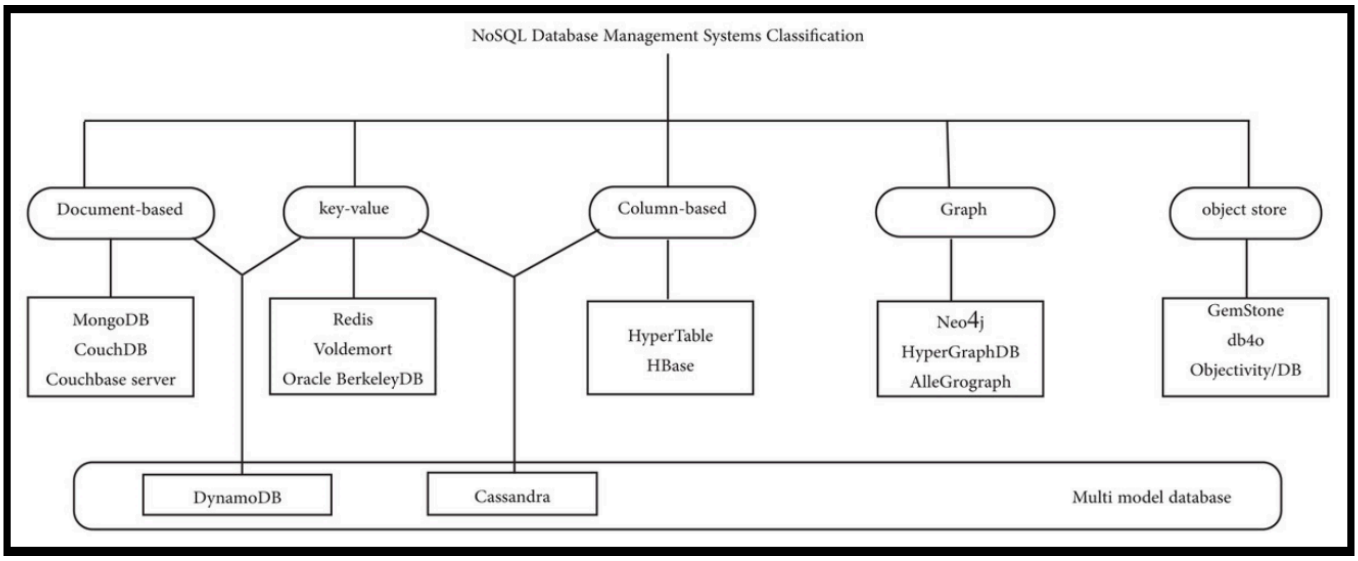
JSON Format

- {}
- Key:value pairs
- Keys must be strings, and values must be a valid JSON data type
  - string,
  - number,
  - object,
  - array,
  - boolean or
  - null



JSON Format

- {}
- Key:value pairs
- Keys must be strings, and values must be a valid JSON data type
- Keys and values are separated by a colon (:).
- Each key/value pair is separated by a comma(,).



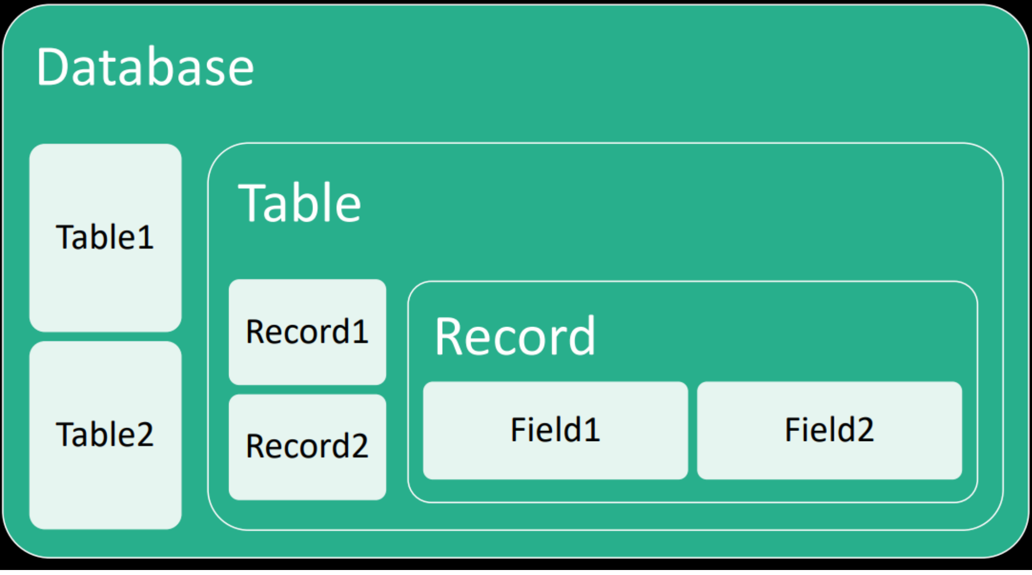
MongoDB

- Document-based
- JSON, BSON, XML based documents

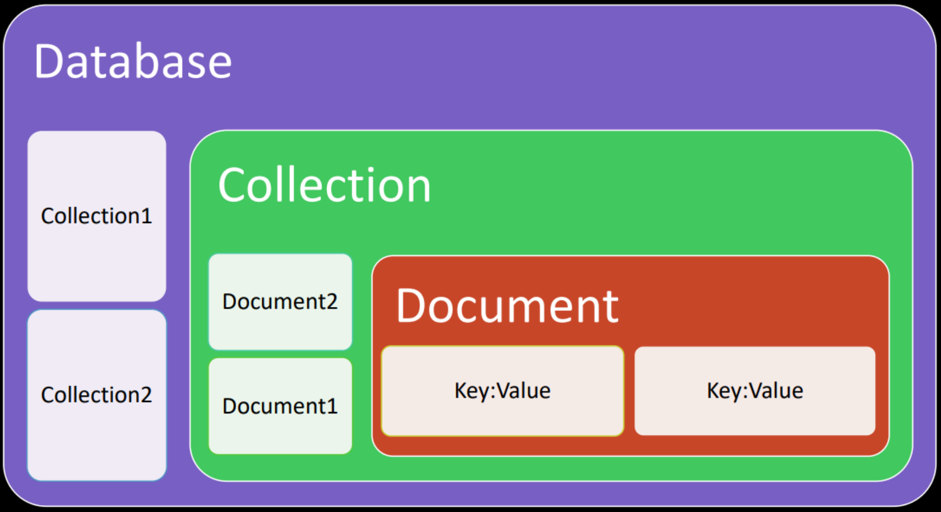
```
{
   field1: value1,
   field2: value2,
   field3: value3,
   ...
   fieldN: valueN
}
```

```
{
   name: "sue",                    ←— field: value
   age: 26,                        ←— field: value
   status: "A",                    ←— field: value
   groups: [ "news", "sports" ]    ←— field: value
}
```

Recap SQL



MongoDB



Point to remember

- Creation of a Database => creation of first collection
- Collections ~ Tables in SQL
- Documents ~ Records in SQL

- Nested Documents

CRUD operations

- C – Create operations

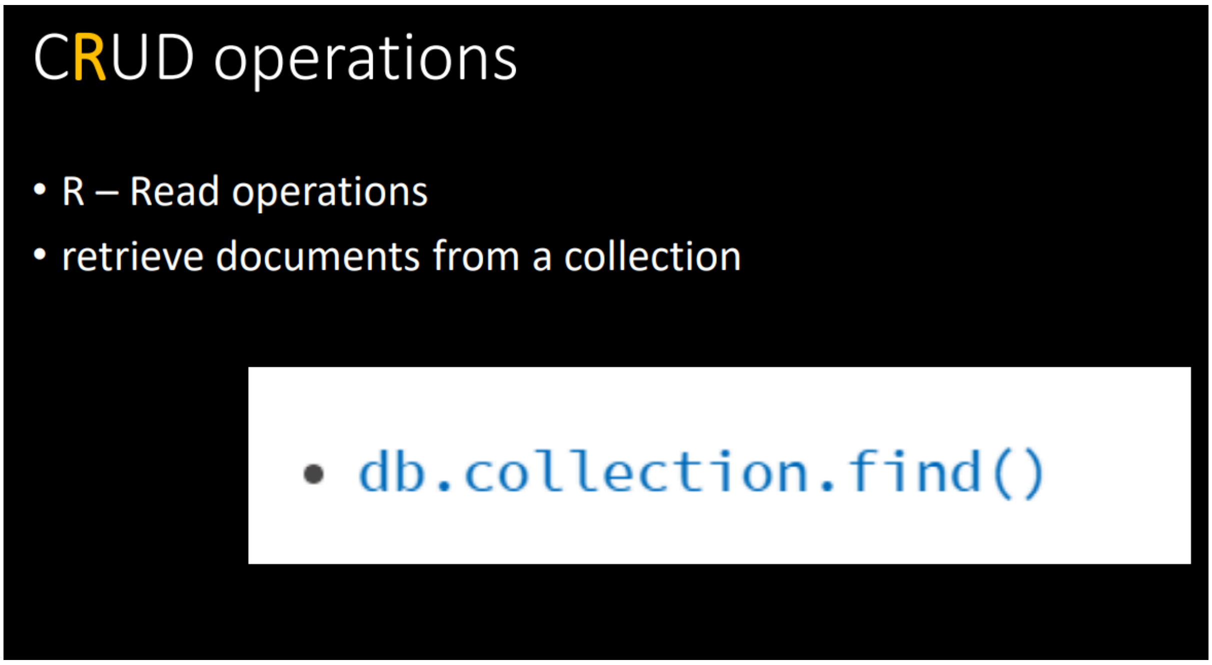
- add new documents to a collection.

- What if no collection exists?
  - Yes, insert operations will create the collection

CRUD operations

- Add new documents
  - Add one
  - Add many

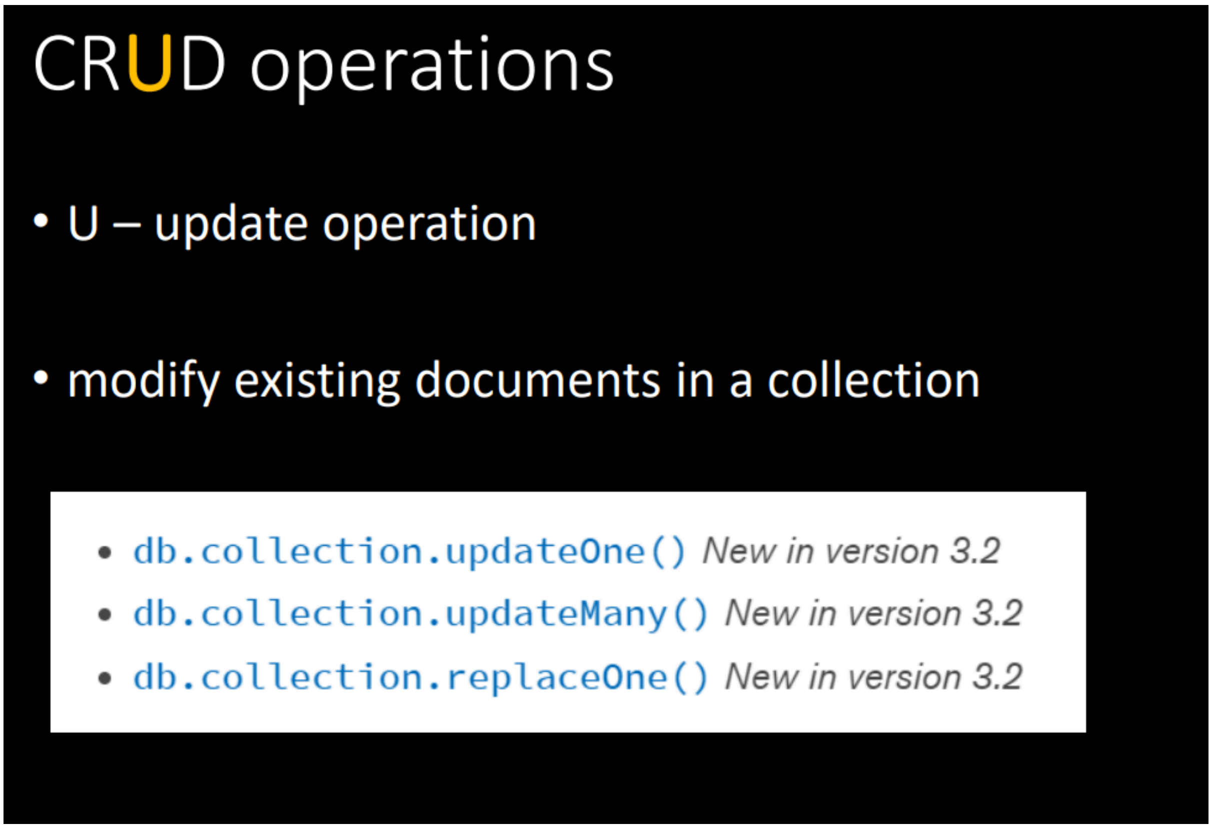
- `db.collection.insertOne()` *New in version 3.2*
- `db.collection.insertMany()` *New in version 3.2*



CRUD operations

- R – Read operations
- retrieve documents from a collection

```
• db.collection.find()
```



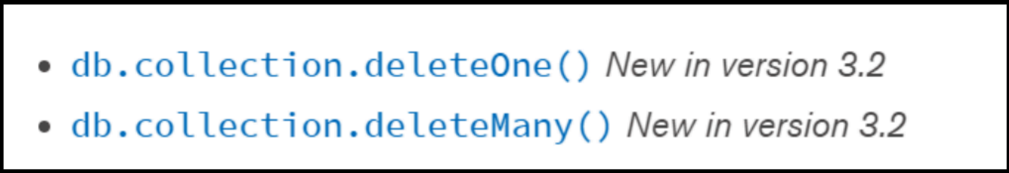
CRUD operations

- U – update operation
- modify existing documents in a collection

```
• db.collection.updateOne() New in version 3.2
• db.collection.updateMany() New in version 3.2
• db.collection.replaceOne() New in version 3.2
```

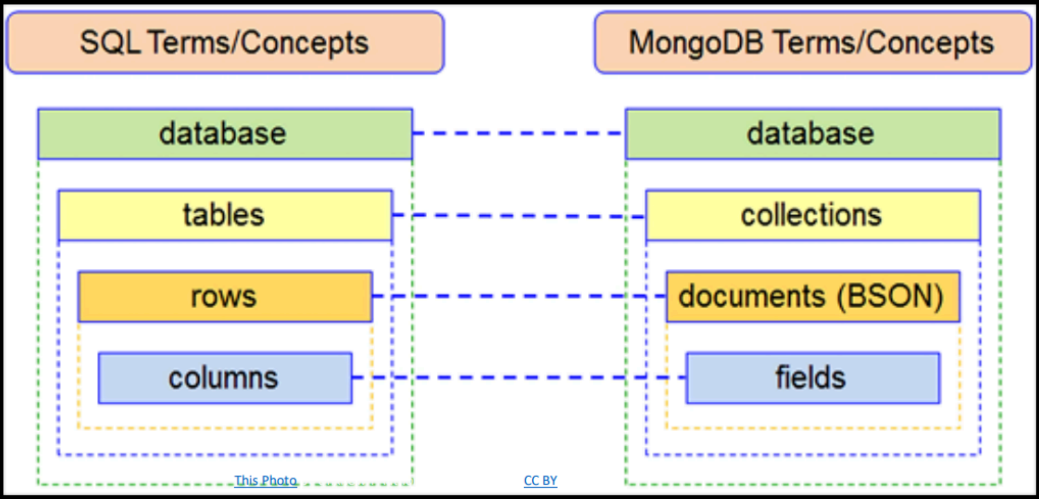
CRUD operations

- D – delete operation
- remove documents from a collection



- `db.collection.deleteOne()` *New in version 3.2*
- `db.collection.deleteMany()` *New in version 3.2*

Recap: SQL vs NoSQL


SQL Terms/Concepts
MongoDB Terms/Concepts
database
database
tables
collections
rows
documents (BSON)
columns
fields