

# Lab 3

# Contents

- Adding elements
- Using arrays as sets
- Removing elements
- Positional array modifications
- Upserts
- Querying
- Project – User Schema

# Adding elements

- **\$push** adds elements to the end of an array.
- You can push multiple values in one operation using the **\$each** suboperator.
- Syntax
  - `update({<query>}, {$push:{<field1>:<value1>,...}})`
  - `update({<query>}, {$push:{<field1>:{$each:[<value1>,<value2>,<value3>,...]}}})`

# Exercise 1

- Insert the document into **posts** collection in lab3.
  - `{"name" : "Lee", "content" : "Hello I'm Lee"}`
- Add a "comments" key containing an array.
  - `{"comments" : {"name" : "Kim", "content" : "Good posts.", "votes" : 0}}`
- Add multiple comments.
  - `{"comments" : {"name" : "Choi", "content" : "How is it going?", "votes" : 1}}`
  - `{"comments" : {"name" : "David", "content" : "I'm David, What's up?", "votes" : 2}}`
  - `{"comments" : {"name" : "Jung", "content" : "Glad to hear that", "votes" : 3}}`

# Using array as sets

- Treat array as a set, only adding values if they are not present, using **\$addToSet** in the query document.
- You can also use **\$addToSet** in conjunction with **\$each** to add multiple unique values.
- Syntax
  - `update({<query>}, {$addToSet : {<field1>:<value1>,...}})`
  - `update({<query>}, {$addToSet : {<field1>:{$each:[<value1>,<value2>,...]}}})`

# Exercise 2



- Insert the document into **users** collection in lab3.
  - {"name" : "Joe", "emails" : ["joe@naver.com", "joe@gmail.com", "joe@yahoo.com"]}
- Add other addresses using **\$addToSet** to prevent duplicates.
  - {"emails" : "joe@gmail.com"}
  - {"emails" : "joe@hotmail.com"}

# Removing elements

- Several ways to remove elements from an array.
- Treat the array like a queue or a stack.
  - Use **\$pop** which can remove elements from either end or beginning.
- Syntax
  - `update({<query>},{ $pop : {<field> : 1}})` # from the end of the array
  - `update({<query>},{ $pop : {<field> : -1}})` # from the beginning of the array

# Removing elements

- Remove an element based on specific criteria.
  - Use **\$pull** to remove the element that match the given criteria.
  - Pulling removes all matching documents, not just a single match.
- Syntax
  - `update({<query>},{ $pull : {<field1> : <value1>}})`



# Exercise 3

- Use the **users** collection in lab3.
- Remove first element of joe's emails using **\$pop**.
- Remove the element "joe@naver.com" of joe's emails using **\$pull**.

# Positional array modification

- Two ways to manipulate values in arrays.
  - By position
  - Position operator (the \$ character)
    - When we don't know what index of the array to modify.
    - Updates only the first match.
  - Example
    - `update(<query>, {$inc : {"comments.0.votes" : 1}})`
    - `update(<query>, {$inc : {"comments.$.votes" : 1}})`

## Exercise 4

- Use the **posts** collection in lab3.
- Set the votes' value of the second comment of Lee's post to 2.
- Set the votes' value of the Kim's comment of Lee's post to 5 using the position operator.

# Upserts

- A special type of update.
- If no document is found, a new document will be created by combining the criteria.
- If a matching document is found, it will be updated normally.
- Specified by the third parameter of update().

# Updating multiple documents



- Updates, by default, update only the first document found that matches the criteria.
- To modify all of the documents, pass **true** as the fourth parameter to update.

# Exercise 5

- Insert documents into people in lab3.
  - `{"name" : "Kim", "age" : 21, "profile" : "Hello I'm Kim"}`,
  - `{"name" : "Lee", "age" : 22 , "profile" : "Hello I'm Lee"}`,
  - `{"name" : "Jung", "age" : 22 , "profile" : "Hello I'm Jung"}`,
  - `{"name" : "Park", "age" : 26 , "profile" : "Hello I'm Park"}`
- Set all the documents' profile value in people collections to "Your account have been hacked!"

# Querying – find() (1)

- An empty query document (i.e. {}) matches everything in the collection.
  - `db.collections.find()`
- Add key/value pairs to restrict the search.
  - `db.collections.find({<field> : <value>})`
- Add more key/value pairs for multiple conditions.
  - `db.collections.find({<field1> : <value1>, <field2> : <value2>})`

# Querying – find() (2)

- Pass a second argument to specify which keys to return/exclude.
  - `db.collections.find({}, {<field1> : 1, <field2> : 1})`
  - `db.collections.find({}, {<field1> : 1, <field2> : 0})` # never want to return <field2>



# Exercise 6

- Use the people collection in lab3.
- Find the document whose age is 22.
- Find the document whose name is Lee except the profile.

# Querying – Conditional queries

- More complex queries criteria such as ranges, OR, and negation.
- Range and not equal queries
  - Use **\$lt**, **\$lte**, **\$gt**, and **\$gte** for comparison queries.
    - `db.users.find({"age" : {$gte : 18, $lte : 30}})`
  - Use **\$ne** for not equal queries.
    - `db.users.find({"username" : {$ne : "joe"}})`

# Exercise 7

- Use the people collection in lab3.
- Find the documents whose age is between 22 and 26.
- Find the documents whose name is not Lee.

# Querying – OR queries

- \$in can be used to query for a variety of values for a single key
  - `db.raffle.find({"ticket_no" : {$in : [725, 542, 390]}})`
- \$nin which is the opposite of \$in returns documents that don't match any of the criteria in the array.
  - `db.raffle.find({"ticket_no" : {$nin : [725,542,390]}})`
- \$or can be used to query for any of the given values across multiple keys.
  - `db.raffle.find({$or : [{"ticket_no" : 725}, {"winner" : true}]})` # ticket\_no is 725 or winner is true

# Exercise 8



- Use the people collection in lab3.
- Find the documents whose name is "Lee" or "Jung".
- Find the documents whose name is not "Kim" or "Park" or "Jung".

# Exercise 9 in PyMongo

- Access the array elements by index
  - `array["key"][i]`
- Assign letter grade for each student based on their total scores (grade.txt).
  - Total score = (mid + final) / 2
  - $\geq 90$  : A
  - $\geq 80$  : B
  - $\geq 70$  : C
  - else : D
- Update the "grade" field for each student.

# User 기능 구현

# Donald Trump's profile page in Twitter



The image is a screenshot of Donald Trump's Twitter profile page. At the top, there is a circular profile picture of Donald Trump and a large red and white wavy banner. Below the profile picture, the name "Donald J. Trump" is displayed with a verified account icon, followed by the handle "@realDonaldTrump". His bio reads "45th President of the United States of America" with a US flag emoji. Location is "Washington, DC" and he "Joined March 2009". There are "2,319 Photos and videos" listed. To the right of the bio, statistics are shown: 36.1K Tweets, 45 Following, 40.6M Followers, 17 Likes, and 5 Moments. Below the bio, there are three small image thumbnails. The main content area shows two tweets. The first tweet says "I was recently asked if Crooked Hillary Clinton is going to run in 2020? My answer was, 'I hope so!'" with 15K replies, 10K retweets, and 41K likes. The second tweet says "The U.S. has gained more than 5.2 trillion dollars in Stock Market Value since Election Day! Also, record business enthusiasm." with 7.0K replies, 6.5K retweets, and 30K likes. Navigation tabs for "Tweets", "Tweets & replies", and "Media" are visible above the tweet list.

**Donald J. Trump** ✓  
@realDonaldTrump

45th President of the United States of America 🇺🇸

Washington, DC

Joined March 2009

2,319 Photos and videos

**Tweets**   **Tweets & replies**   **Media**

**Donald J. Trump** ✓ @realDonaldTrump · 2h  
I was recently asked if Crooked Hillary Clinton is going to run in 2020? My answer was, "I hope so!"  
15K   10K   41K

**Donald J. Trump** ✓ @realDonaldTrump · 2h  
The U.S. has gained more than 5.2 trillion dollars in Stock Market Value since Election Day! Also, record business enthusiasm.  
7.0K   6.5K   30K



# User 관련 기능

- 계정 생성시 유저아이디, 이름, 비밀번호를 입력 받는다.  
(비밀번호 확인 기능도 구현)
- 각 계정은 계정정보 뿐만 아니라 상태메시지와 팔로잉/팔로워 목록을 가지고 있다.
- My Status 에서 자신의 상태 메시지, 팔로워 수, 팔로잉 수를 출력한다.

# Today, you will implement ...

- main.py
- user.py
- 더미 코드를 다운로드 받아서, 주식 부분을 작성해주세요.

# Test your program

- 각 메소드가 제대로 동작하는지?
- 삽입된 컬렉션과 다크먼트들이 db에 잘 저장되었는지?
- 예외처리가 잘 되었는지?