HW3_template

October 30, 2024

[1]:
```
# Install pymongo if applicable
!python -m pip install "pymongo[srv]"
```

```
Collecting pymongo[srv]
  Downloading pymongo-4.10.1-cp312-cp312-macosx_11_0_arm64.whl.metadata (22 kB)
WARNING: pymongo 4.10.1 does not provide the extra 'srv'

Collecting dnspython<3.0.0,>=1.16.0 (from pymongo[srv])
  Downloading dnspython-2.7.0-py3-none-any.whl.metadata (5.8 kB)
Downloading dnspython-2.7.0-py3-none-any.whl (313 kB)
Downloading pymongo-4.10.1-cp312-cp312-macosx_11_0_arm64.whl (943 kB)

943.1/943.1 kB 8.5 MB/s eta 0:00:00
Installing collected packages: dnspython, pymongo
Successfully installed dnspython-2.7.0 pymongo-4.10.1
```

[2]:
```
# setup the client. Make sure to remove your password while submitting
import pymongo
from bson.json_util import dumps

uri = "mongodb+srv://cwang131:#@cs224p.s5z8x.mongodb.net/?
 ↪retryWrites=true&w=majority&appName=CS224P"
client = pymongo.MongoClient(uri)
zotmusic = client.zotmusic
```

[3]:
```
# Use Python comments to answer Q1 below
# 1.A. In the records collection, the data type for field release_by is␣
 ↪document. release_by is a nested filed consist of
#      artist_user_id (string) and release_date (string).

# 1.B. Yes, there are one field hat is not present in the original data file␣
 ↪(records.json)
#    i. The name of the newly added field is _id.
#    ii. The data type of _id is ObjectId.
#    iii. The _id field in MongoDB, specifically when it's of type ObjectId, is␣
 ↪a unique identifier that MongoDB automatically assigns
#         to each document if you don't specify one yourself. It is a 12-byte␣
 ↪identifier that MongoDB uses as a primary key for each document,
```

```
#         ensuring uniqueness within the collection.
#    iv. Yes, the _id field is unique for all documents in a MongoDB collection.
 ↪

# 1.C. In the reviews collection, posted_by.user_id field can be used to␣
 ↪identify the creator of a review by linking it with the users collection.

# 1.D. In the users collection, the data type for the field subscription is␣
 ↪string. The values are monthly, free, yearly.

# 1.E. In the users collection, the data type for the field genres is array of␣
 ↪string values.The average of genres per user is 4.62 which means
#      that a user maintains around 5 genres on average.
```

[15]:
```
# 2A
cursor = zotmusic.records.find_one({"title": "Big worry"})
print(dumps(cursor, indent=2))
```


```
{
  "_id": {
    "$oid": "67215bbe18de91b7d0489282"
  },
  "record_id": "record_HPJtcrZW",
  "title": "Big worry",
  "genre": "Reggae",
  "released_by": {
    "artist_user_id": "user_YEGFl37i",
    "release_date": "2020-11-15"
  },
  "is_album": false,
  "is_single": true,
  "video_url": "http://walker-sanchez.org/",
  "songs": [
    {
      "track_number": 1,
      "title": "Big worry",
      "length": 435,
      "bpm": 142,
      "mood": "people"
    }
  ]
}
```


[17]:
```
# 2B
cursor = zotmusic.records.find({"is_album": True}, {"_id": 0, "title": 1,␣
 ↪"released_by.release_date": 1}).sort("released_by.release_date", -1).limit(3)
print(dumps(cursor, indent=2))
```

```
[
  {
    "title": "Position option hair",
    "released_by": {
      "release_date": "2023-12-29"
    }
  },
  {
    "title": "Whose hear",
    "released_by": {
      "release_date": "2023-12-28"
    }
  },
  {
    "title": "House film",
    "released_by": {
      "release_date": "2023-12-24"
    }
  }
]
```


[18]:
```
# 2C
cursor = zotmusic.users.count_documents({"is_listener": True, "is_artist":␣
 ↪True})
print(cursor)
```

100

[20]:
```
# 2D
cursor = zotmusic.records.count_documents({
    "$or": [
        {"is_single": True, "genre": "Pop", "released_by.release_date": {"$gt":␣
 ↪"2023-06-30"}},
        {"is_album": True, "genre": "Jazz", "released_by.release_date": {"$lt":␣
 ↪"2023-01-01"}}
    ]
})
print(cursor)
```

24

[49]:
```
# 2E
cursor = zotmusic.users.find(
    {
        "is_listener": True,
        "$expr": {"$gte": [{"$size": "$genres"}, 9]},
        "address.street": {"$exists": True, "$ne": None},
        "address.city": {"$exists": True, "$ne": None},
```

```
        "address.state": {"$exists": True, "$ne": None},
        "address.zip": {"$exists": True, "$ne": None}
    },
    {
        "_id": 0,
        "email": 1,
        "full_name": {"$concat": ["$real_name.last_name", ", ", "$real_name.
 ↪first_name"]}
    }
).sort("email", 1)

print(dumps(cursor, indent=2))
```

```
[
  {
    "email": "courtney36@protonmail.com",
    "full_name": "Sherman, Jason"
  },
  {
    "email": "ryanmorgan@icloud.com",
    "full_name": "Wood, Kimberly"
  }
]
```

[6]:
```
# 2F
match_stage = {
    "$match": {
        "is_listener": True,
        "subscription": "yearly",
        "joined_date": {
            "$gte": "2022-01-01",
            "$lt": "2023-01-01"
        }
    }
}

count_stage = {
    "$count": "total_listeners"
}

pipeline = [match_stage, count_stage]

cursor = zotmusic.users.aggregate(pipeline)
print(dumps(cursor, indent=2))
```

```
[
  {
    "total_listeners": 11
```

```
  }
]
```

[8]:
```
# 2G

group_stage = {
    "$group": {
        "_id": "$posted_by.user_id",
        "total_reviews": {"$sum": 1}
    }
}

sort_stage = {
    "$sort": {
        "total_reviews": -1
    }
}

windowfield_stage = {
    "$setWindowFields": {
        "partitionBy": None,
        "sortBy": { "total_reviews": -1 },
        "output": {
            "dense_rank": {
                "$denseRank": {}
            }
        }
    }
}

match_stage = {
    "$match": {
        "dense_rank": { "$lte": 4 }
    }
}

project_stage = {
    "$project": {
        "_id": 0,
        "user_id": "$_id",
        "total_reviews": 1,
        "dense_rank": 1
    }
}

pipeline = [group_stage, sort_stage, windowfield_stage, match_stage,␣
 ↪project_stage]
```

```
cursor = zotmusic.reviews.aggregate(pipeline)
print(dumps(cursor, indent=2))
```


```
[
  {
    "total_reviews": 74,
    "dense_rank": 1,
    "user_id": "user_ux9_qRS-"
  },
  {
    "total_reviews": 71,
    "dense_rank": 2,
    "user_id": "user_vu4RI4dq"
  },
  {
    "total_reviews": 69,
    "dense_rank": 3,
    "user_id": "user_Z217vp0p"
  },
  {
    "total_reviews": 68,
    "dense_rank": 4,
    "user_id": "user_V3iJ0OgE"
  },
  {
    "total_reviews": 68,
    "dense_rank": 4,
    "user_id": "user_ODWOljEV"
  }
]
```


[9]:
```
# 2H
match_stage_1 = {
    "$match": {
        "record_id": "record_qGHboIDL",
        "rating": { "$gte": 4 }
    }
}

lookup_stage = {
    "$lookup": {
        "from": "users",
        "localField": "posted_by.user_id",
        "foreignField": "user_id",
        "as": "user_info"
    }
```

```
}

unwind_stage = {
    "$unwind": "$user_info"
}

match_stage_2 = {
    "$match": {
        "user_info.address.street": { "$exists": True, "$ne": None },
        "user_info.address.city": { "$exists": True, "$ne": None },
        "user_info.address.state": { "$exists": True, "$ne": None },
        "user_info.address.zip": { "$exists": True, "$ne": None }
    }
}

project_stage = {
    "$project": {
        "_id": 0,
        "user_id": "$user_info.user_id",
        "nickname": "$user_info.nickname",
        "address": "$user_info.address"
    }
}

pipeline = [match_stage_1, lookup_stage, unwind_stage, match_stage_2,␣
 ↪project_stage]
cursor = zotmusic.reviews.aggregate(pipeline)
print(dumps(cursor, indent=2))
```


```
[
  {
    "user_id": "user_p4EheoPI",
    "nickname": "ellengibson",
    "address": {
      "street": "9400 Wu Court",
      "city": "New Andre",
      "state": "New Hampshire",
      "zip": "95877"
    }
  },
  {
    "user_id": "user_BerqqJn3",
    "nickname": "davidchad",
    "address": {
      "street": "4174 Michele Gardens Suite 245",
      "city": "Ortizbury",
      "state": "Delaware",
      "zip": "53945"
```

```
    }
  }
]
```

[10]:
```
# 2I
lookup_stage = {
    "$lookup": {
        "from": "reviews",
        "localField": "record_id",
        "foreignField": "record_id",
        "as": "reviews"
    }
}

add_fields_stage = {
    "$addFields": {
        "average_rating": { "$avg": "$reviews.rating" }
    }
}

match_stage = {
    "$match": {
        "average_rating": 4.0
    }
}

unwind_stage = {
    "$unwind": "$songs"
}

bpm_match_stage = {
    "$match": {
        "songs.bpm": { "$lt": 63 }
    }
}

project_stage = {
    "$project": {
        "_id": 0,
        "record_id": 1,
        "record_title": "$title",
        "track_number": "$songs.track_number",
        "song_title": "$songs.title"
    }
}

sample_stage = {
```

```
    "$sample": { "size": 5 }
}

pipeline = [lookup_stage, add_fields_stage, match_stage, unwind_stage,␣
 ↪bpm_match_stage, project_stage, sample_stage]

cursor = zotmusic.records.aggregate(pipeline)
print(dumps(cursor, indent=2))
```


```
[
  {
    "record_id": "record_g6asf3ux",
    "record_title": "Garden approach",
    "track_number": 1,
    "song_title": "Try both whole"
  },
  {
    "record_id": "record_ZBXcXXcq",
    "record_title": "Forward score matter",
    "track_number": 1,
    "song_title": "Note career"
  },
  {
    "record_id": "record_D0VTjwgg",
    "record_title": "Material season",
    "track_number": 8,
    "song_title": "Well peace"
  },
  {
    "record_id": "record_PzgNFItj",
    "record_title": "Develop between prove",
    "track_number": 9,
    "song_title": "System oil"
  },
  {
    "record_id": "record_AxTzPFAx",
    "record_title": "Gas risk",
    "track_number": 9,
    "song_title": "Throw fall"
  }
]
```


[24]:
```
# 2J
lookup_sessions_stage = {
    "$lookup": {
        "from": "sessions",
        "localField": "user_id",
```

```
        "foreignField": "user_id",
        "as": "sessions"
    }
}

unwind_sessions_stage = {
    "$unwind": "$sessions"
}

lookup_records_stage = {
    "$lookup": {
        "from": "records",
        "localField": "sessions.song.record_id",
        "foreignField": "record_id",
        "as": "records"
    }
}

unwind_records_stage = {
    "$unwind": "$records"
}

match_conditions_stage = {
    "$match": {
        "records.released_by.release_date": { "$gt": "2023-01-01" },
        "sessions.replay_count": { "$gt": 3 },
        "sessions.device": "mobile-app"
    }
}

group_by_user_stage = {
    "$group": {
        "_id": {
            "user_id": "$user_id",
            "nickname": "$nickname",
            "email": "$email",
            "joined_date": "$joined_date"
        },
        "distinct_genres": { "$addToSet": "$records.genre" }
    }
}

add_genre_count_stage = {
    "$addFields": {
        "genre_count": { "$size": "$distinct_genres" }
    }
}
```

```
having_genre_count_stage = {
    "$match": {
        "genre_count": { "$gt": 6 }
    }
}

sort_stage = {
    "$sort": {
        "_id.joined_date": -1
    }
}

limit_stage = {
    "$limit": 2
}

project_stage = {
    "$project": {
        "_id": 0,
        "nickname": "$_id.nickname",
        "email": "$_id.email"
    }
}

pipeline = [lookup_sessions_stage, unwind_sessions_stage, lookup_records_stage,␣
 ↪unwind_records_stage, match_conditions_stage,
           group_by_user_stage, add_genre_count_stage,␣
 ↪having_genre_count_stage, sort_stage, limit_stage, project_stage]

cursor = zotmusic.users.aggregate(pipeline)
print(dumps(cursor, indent=2))
```


```
[
  {
    "nickname": "prhodes",
    "email": "prhodes@foxmail.com"
  },
  {
    "nickname": "gomezbrittany",
    "email": "gomezbrittany@foxmail.com"
  }
]
```