Applicazione Python "Premi Nobel"

Attività scelta: 3. MongoDB Usecase e applicazione Python
Progetto realizzato da Elena Curti con Python 3.9.13.

Ho usato i seguenti file JSON pubblici:

- nobelPrizes.json: https://masterdataapi.nobelprize.org/2.1/nobelPrizes?offset=0&limit=664
- laureates.json: https://masterdataapi.nobelprize.org/2.1/laureates?offset=0&limit=981

Essi contengono informazioni sui premi Nobel ed i relativi vincitori.

Documentazione

Documentazione dei dataset: https://app.swaggerhub.com/apis/NobelMedia/NobelMasterData/2.1

File con i premi Nobel

Il file *JSON_Files/modificati/nobelPrizes2.json* contiene l'elenco dei premi Nobel, con la seguente struttura d'esempio:

```
{
    "awardYear": 2021,
    "category": "Economic Sciences",
    "categoryFullName": "The Sveriges Riksbank Prize in Economic
Sciences in Memory of Alfred Nobel",
    "dateAwarded": "2021-10-11",
    "topMotivation": "for contributions ...",
    "prizeAmount": 10000000,
    "laureates": [
        {
            "id": "1007",
            "knownName": "David Card", // Solo per le persone
            "orgName": "Institute of International Law", // Solo per
le organizzazioni
            "portion": "1/2",
            "sortOrder": "1",
            "motivation": "for his empirical contributions to labour
economics"
        },
```

```
        { ... }
    ]
}
```

Modifiche effettuate

Rispetto al file JSON originale (contenuto in *JSON_Files/originali/nobelPrizes.json*):

- Ho rimosso i sottoalberi "/link" e "/meta" perchè contenenti informazioni non inerenti ai premi Nobel.
- Ho rimosso la traduzione di alcuni campi. La struttura originale di tali campi era la seguente:

```
"nome": {
    "en": "termine_inglese",
    "no": "termine_svedese",
    "se": "termine_norvegese"
}
```

  Ho deciso di lasciare solo il termine inglese, trasformando quindi i campi in formato testuale. Ho eseguito questa modifica per i campi: "category", "categoryFullName", "laureates.knownName" e "laureates.motivation".
- Ho trasformato "awardYear" in un campo intero (originariamente testuale)

File con i vincitori

Per arricchire il database, ho deciso di inserire anche il file *JSON_Files/modificati/laureates2.json* contiene l'elenco dei vincitori dei premi Nobel, con la seguente struttura d'esempio:

```
// Per le persone
{
    "id": "102",
    "knownName": "Aage N. Bohr",
    "givenName": "Aage N.",
    "familyName": "Bohr",
    "fullName": "Aage Niels Bohr",
    "gender": "male",
    "birth": {
        "date": "1922-06-19",
        "place": {
```

```
            "cityNow": "Copenhagen",
            "countryNow": "Denmark"
        }
    },
    "death": {
        "date": "2009-09-08",
        "place": {
            "cityNow": "Copenhagen",
            "countryNow": "Denmark"
        }
    },
    "wikipedia": "https://en.wikipedia.org/wiki/Aage_Bohr"
}

// Per le organizzazioni
{
    "id": "537",
    "orgName": "Amnesty International",
    "nativeName": "Amnesty International",
    "founded": {
        "date": "1961-00-00",
        "place": {
            "cityNow": "London",
            "countryNow": "United Kingdom"
        }
    },
    "wikipedia":
"https://en.wikipedia.org/wiki/Amnesty_International"
}
```

Modifiche effettuate

Rispetto al file JSON originale (contenuto in *JSON_Files/originali/laureates.json*):

- Ho rimosso i sottoalberi "/link" e "/meta" perchè contenenti informazioni non inerenti ai vincitori.
- Come prima, ho rimosso la traduzione svedese e norvegese (lasciando solo quella inglese) dei seguenti campi: "knownName", "givenName", "givenName", "fullName", "cityNow", "countryNow", "orgName"
- Ho modificato i campi "birth.place", "death.place" e "founded.place" rimuovendone i sotto-campi "city", "country", "continent" e "locationString". In questo modo ho semplificato la

struttura del file, rimuovendo informazioni molto specifiche sui luoghi geografici e non inerenti al vincitore del premio.
- Ho rimosso il campo "nobelPrizes", contenente l'elenco dei premi Nobel vinti dalla persona/organizzazione, perchè già memorizzati nel file nobelPrizes2.json.

Struttura delle Collection

Nelle interrogazioni proposte nell'applicazione, ho dato la possibilità all'utente di scegliere se visionare i vincitori dei premi in modo sintetico (solo il nome) o dettagliato (tutti i campi). Quindi, ho deciso di non unire i file laureates2.json e nobelPrizes2.json, e di tenere due collections distinte.

Applicazione

Eseguire il codice nelle celle, seguendo l'ordine proposto.

Operazioni iniziali e definizione di funzioni

Il seguente codice crea ed inizializza un database chiamato *premi_nobel*, formato dalle due collections *laureates2* e *nobelPrizes2* (contenenti i dati memorizzati nei relativi file JSON).
NB: Eseguedo questo codice, gli eventuali dati memorizzati nelle collections e non presenti nei file JSON saranno eliminati. Eseguire quindi il codice nel primo avvio dell'applicazione o quando si vuole ripristinare il database.

In [1]:
```
import json, sys, os
from pymongo import MongoClient

class MiaStopExecution(Exception):
    """Il raise di questa classe provoca l'interruzione dell'esecizione della cella, senza interrompere il kernel"""
    def _render_traceback_(self):
        pass

def carica_dati(nomeFile, collection):
    ''' Funzione che carica il file json nella collection. '''
    collection.drop()
    try:
        with open(path_cartella_file_json+nomeFile, encoding='utf8') as file:
            file_data = json.load(file)
    except FileNotFoundError:
        print("Path inserito non valido! Assicurarsi di aver messo / o \\\\ alla fine del path.", file=sys.stderr)
```

```
        raise MiaStopExecution
    collection.insert_many(file_data)

# Costanti
FILE_LAUREATES = "laureates2.json"
FILE_NOBEL_PRICES = "nobelPrizes2.json"
LOOKUP_JOIN_COLLECTIONS = {
            '$lookup': {
                "localField": "laureates.id",
                "from": "laureates2",
                "foreignField": "id",
                "as": "laureates"
            }
        }

# Inizializzazione del database
myclient = MongoClient("mongodb://localhost:27017/")
db = myclient["premi_nobel"]
collection_laureates2 = db["laureates2"]
collection_nobelPrices2 = db["nobelPrizes2"]

path_cartella_file_json =  "JSON_Files" + os.sep + "modificati" + os.sep

carica_dati(FILE_LAUREATES, collection_laureates2)
carica_dati(FILE_NOBEL_PRICES, collection_nobelPrices2)
print("Database inizializzato correttamente")
```

```
Database inizializzato correttamente
```

In [9]:

```
def get_info_event(dizionario):
    """Funzione che ritorna l'evento in input serializzato in una stringa"""
    ret_str = ""
    if "date" in dizionario:
        ret_str += dizionario["date"]
    if "place" in dizionario:
        ret_str += " a "+ dizionario["place"].get("cityNow", "") + ", "
+dizionario["place"].get("countryNow", "")
    return ret_str


def print_vincitore(l, stampa_dettagli):
```

```
    """Funzione che stampa i dati del vincitore in input"""
    if "knownName" in l:
        print("Persona: " + str(l["knownName"]))
    if "orgName" in l:
        print("Organizzazione: " + str(l["orgName"]))

    if not stampa_dettagli:
        return
    # Dettagli persone
    if "givenName" in l or "familyName" in l or "fullName" in l:
        print("\tAltri nomi: " + str(l.get("fullName", '')) + " - " +
str(l.get("givenName", '')) +" - " + str(l.get("familyName", '')))
    if "gender" in l:
        print("\tSesso: " + str(l["gender"])[0])
    if "birth" in l:
        print("\tNato/a il " + get_info_event(l["birth"]))
    if "death" in l:
        print("\tMorto/a il " + get_info_event(l["death"]))

    # Dettagli organizzazioni
    if "nativeName" in l:
        print("\tAltri nomi: " + str(l["nativeName"]))
    if "founded" in l:
        print("\tFondato/a il " + get_info_event(l["founded"]))

    # Dettagli comuni
    if "wikipedia" in l:
        print("\tWikipedia: " + str(l["wikipedia"]))

def print_nobels(nobel_list, stampa_dettagli):
    """Funzione che stampa il nobel dato in input (in modo dettagliato se
stampa_dettagli e' true)."""

    def print_nobel(nobel):
        print("Anno: " + str(nobel.get('awardYear', "-")))
        print("Categoria: " + str(nobel.get('category', "-")) + " - anche
detta " + str(nobel.get('categoryFullName', "")))
        print("Data: " + str(nobel.get('dateAwarded', "-")))
        print("Motivazione: " + str(nobel.get('topMotivation', "-")))
        if "prizeAmount" in nobel:
```

```
            prizeAmount = nobel['prizeAmount']
            print("Importo del premio (in corone svedesi): " +
get_importo(prizeAmount))
        laureates = nobel.get('laureates', [])
        for l in laureates:
            print_vincitore(l, stampa_dettagli)
            if 'portion' in l:
                print("\tPorzione del Nobel vinta: " +  str(l['portion']))
            if 'motivation' in l:
                print("\tMotivazione: " +  str(l['motivation']))


    for nobel in nobel_list:
        print("**************")
        print_nobel(nobel)

def chiedi_stampa_dettagli():
    return input("Stampare anche i dettagli dei vincitori
(y/n)?").lower()=='y'

def get_importo(importo):
    return str(f"{importo:,.2f}")

def chiedi_categoria():
    print("Scegli una categoria tra: ")
    lista_categorie = collection_nobelPrices2.distinct("category")
    i = 0
    for cat in lista_categorie:
        print(str(i)+ " -- " + cat)
        i+=1
    try:
        num_categoria = int(input(">> "))
        if num_categoria <0 or num_categoria >= len(lista_categorie):
            print("Valore non valido!")
            raise MiaStopExecution

        scelta = lista_categorie[num_categoria]

    except ValueError:
        print("Valore non valido!")
```

```
        raise MiaStopExecution
    return scelta
```

Interrogazioni

1. Cercare tutti i vincitori dei premi Nobel di una categoria

In [10]:

```
print("Cercare tutti i vincitori dei premi Nobel di una categoria")

# Stampa del menu e richiesta degli input
print("Quale categoria? ")
categoria = chiedi_categoria()

stampa_dettagli = input("Stampare anche i dettagli dei vincitori
(y/n)?").lower()=='y'
print("Scelta: " +  categoria+ " "+("con" if stampa_dettagli else "senza" ) +
" dettagli")

# Interrogazione e stampa dei risultati
if stampa_dettagli:
    lista_nobel = collection_nobelPrices2.aggregate([
        {'$match': {"category":categoria}},
        LOOKUP_JOIN_COLLECTIONS
    ])
else:
    lista_nobel = collection_nobelPrices2.find({"category":categoria})

print_nobels(lista_nobel, stampa_dettagli)
```

```
Cercare tutti i vincitori dei premi Nobel di una categoria
Quale categoria?
Scegli una categoria tra:
0 -- Chemistry
1 -- Economic Sciences
2 -- Literature
3 -- Peace
4 -- Physics
5 -- Physiology or Medicine
>> 3
Stampare anche i dettagli dei vincitori (y/n)?n
Scelta: Peace senza dettagli
*************
Anno: 1901
Categoria: Peace - anche detta The Nobel Peace Prize
Data: 1901-12-10
Motivazione: -
Importo del premio (in corone svedesi): 150,782.00
Persona: Henry Dunant
        Porzione del Nobel vinta: 1/2
        Motivazione: for his humanitarian efforts to help wounded soldiers and create int
ernational understanding
Persona: Frédéric Passy
        Porzione del Nobel vinta: 1/2
        Motivazione: for his lifelong work for international peace conferences, diplomacy
and arbitration
*************
Anno: 1902
Categoria: Peace - anche detta The Nobel Peace Prize
Data: 1902-12-10
Motivazione: -
Importo del premio (in corone svedesi): 141,847.00
Persona: Élie Ducommun
        Porzione del Nobel vinta: 1/2
        Motivazione: for his untiring and skilful directorship of the Bern Peace Bureau
Persona: Albert Gobat
        Porzione del Nobel vinta: 1/2
        Motivazione: for his eminently practical administration of the Inter-Parliamentar
y Union
*************
Anno: 1903
Categoria: Peace - anche detta The Nobel Peace Prize
Data: 1903-12-10
Motivazione: -
Importo del premio (in corone svedesi): 141,358.00
Persona: Randal Cremer
        Porzione del Nobel vinta: 1
        Motivazione: for his longstanding and devoted effort in favour of the ideas of pe
ace and arbitration
*************
Anno: 1904
Categoria: Peace - anche detta The Nobel Peace Prize
Data: 1904-12-10
Motivazione: -
Importo del premio (in corone svedesi): 140,859.00
Organizzazione: Institute of International Law
```

Nella stampa del PDF questo output e' stato interrotto.

2. Cercare un Nobel (con data e anno)

In [11]:

```
print("Cercare un Nobel (con data e anno)")

# Richiesta degli input
anno = int(input("Inserire anno:"))
categoria = chiedi_categoria()
stampa_dettagli = input("Stampare anche i dettagli dei vincitori
(y/n)?").lower()=='y'
print("Scelta: Anno: "+str(anno) +  "\tCategoria: " + categoria+ "\t"+("con"
if stampa_dettagli else "senza" ) + " dettagli")

# Interrogazione e stampa dei risultati
if stampa_dettagli:
    lista_nobel = collection_nobelPrices2.aggregate([
        {'$match': {'awardYear':anno, "category":categoria}},
        LOOKUP_JOIN_COLLECTIONS
    ])
else:
    lista_nobel =
collection_nobelPrices2.find({'awardYear':anno,"category":categoria})

print_nobels(lista_nobel, stampa_dettagli)
```

```
Cercare un Nobel (con data e anno)
Inserire anno:2000
Scegli una categoria tra:
0 -- Chemistry
1 -- Economic Sciences
2 -- Literature
3 -- Peace
4 -- Physics
5 -- Physiology or Medicine
>> 0
Stampare anche i dettagli dei vincitori (y/n)?y
Scelta: Anno: 2000      Categoria: Chemistry    con dettagli
*************
Anno: 2000
Categoria: Chemistry - anche detta The Nobel Prize in Chemistry
Data: 2000-10-10
Motivazione: -
Importo del premio (in corone svedesi): 9,000,000.00
Persona: Alan Heeger
        Altri nomi: Alan J. Heeger - Alan - Heeger
        Sesso: m
        Nato/a il 1936-01-22 a Sioux City, IA, USA
        Wikipedia: https://en.wikipedia.org/wiki/Alan_J._Heeger
Persona: Alan MacDiarmid
        Altri nomi: Alan G. MacDiarmid - Alan - MacDiarmid
        Sesso: m
        Nato/a il 1927-04-14 a Masterton, New Zealand
        Morto/a il 2007-02-07 a Drexel Hill, PA, USA
        Wikipedia: https://en.wikipedia.org/wiki/Alan_MacDiarmid
Persona: Hideki Shirakawa
        Altri nomi: Hideki Shirakawa - Hideki - Shirakawa
        Sesso: m
        Nato/a il 1936-08-20 a Tokyo, Japan
        Wikipedia: https://en.wikipedia.org/wiki/Hideki_Shirakawa
```

3. Stampare i primi Nobel in ordine (crescente o decrescente) di importo o data

In [12]:

```python
print("Stampare i primi N Nobel degli in ordine (crescente o decrescente) di importo o di data ")

# Richiesta degli input
n_str = input("Quanti nobel stampare (premere invio per stamparli tutti)?")
try:
    if n_str!="":
        n = int(n_str if n_str != "" else -1)
        if n<1:
            raise ValueError
    else:
        n = -1
except ValueError:
    print("Errore! Numero inserito non valido")
```

```
    raise MiaStopExecution

importo_o_data=""
while(not (importo_o_data=="1" or importo_o_data=="2")):
    importo_o_data = input("Si desiderano vedere i nobel in ordine
di:\n1.Importo del premio\n2.Data\n>> ")
importo_o_data = "prizeAmount" if importo_o_data == "1" else "dateAwarded"

ordine=""
while(not (ordine=="1" or ordine=="2")):
    ordine = input("Si desiderano vedere i nobel in
ordine:\n1.Crescente\n2.Decrescente\n>> ")
ordine = 1 if ordine == "1" else -1

stampa_dettagli = chiedi_stampa_dettagli()

print("Hai scelto: " +
    ("tutti i nobel" if n==-1 else "primi " + n_str + " nobel" )
    + " in ordine " + ("crescente" if ordine == 1 else "decrescente") + " di "
    + ("data" if importo_o_data == "dateAwarded" else "importo") + " "
    + ("con" if stampa_dettagli else "senza" ) + " dettagli"
)

# Interrogazione e stampa dei risultati
if stampa_dettagli:
    lista_filtri = [
        { '$match': {importo_o_data:{'$exists':'true'}}},
        { '$sort':  {importo_o_data: ordine } },
        LOOKUP_JOIN_COLLECTIONS
    ]
    if n!=-1:
        lista_filtri += [{'$limit':n}]
    lista_nobel = collection_nobelPrices2.aggregate(lista_filtri)
else:
    lista_nobel = collection_nobelPrices2.find({importo_o_data:
{'$exists':'true'}}).sort(importo_o_data, ordine)
    if n != -1:
        lista_nobel = lista_nobel.limit(n)

print_nobels(lista_nobel, stampa_dettagli)
```

```
Stampare i primi N Nobel degli in ordine (crescente o decrescente) di importo o di data
Quanti nobel stampare (premere invio per stamparli tutti)?2
Si desiderano vedere i nobel in ordine di:
1.Importo del premio
2.Data
>> 2
Si desiderano vedere i nobel in ordine:
1.Crescente
2.Decrescente
>> 1
Stampare anche i dettagli dei vincitori (y/n)?y
Hai scelto: primi 2 nobel in ordine crescente di data con dettagli
*************
Anno: 1901
Categoria: Peace - anche detta The Nobel Peace Prize
Data: 1901-12-10
Motivazione: -
Importo del premio (in corone svedesi): 150,782.00
Persona: Frédéric Passy
        Altri nomi: Frédéric Passy - Frédéric - Passy
        Sesso: m
        Nato/a il 1822-05-20 a Paris, France
        Morto/a il 1912-06-12 a Paris, France
        Wikipedia: https://en.wikipedia.org/wiki/Frédéric_Passy
Persona: Henry Dunant
        Altri nomi: Jean Henry Dunant - Henry - Dunant
        Sesso: m
        Nato/a il 1828-05-08 a Geneva, Switzerland
        Morto/a il 1910-10-30 a Heiden, Switzerland
        Wikipedia: https://en.wikipedia.org/wiki/Henry_Dunant
*************
Anno: 1902
Categoria: Peace - anche detta The Nobel Peace Prize
Data: 1902-12-10
Motivazione: -
Importo del premio (in corone svedesi): 141,847.00
Persona: Albert Gobat
        Altri nomi: Charles Albert Gobat - Albert - Gobat
        Sesso: m
        Nato/a il 1843-05-21 a Tramelan, Switzerland
        Morto/a il 1914-03-16 a Bern, Switzerland
        Wikipedia: https://en.wikipedia.org/wiki/Charles_Albert_Gobat
Persona: Élie Ducommun
        Altri nomi: Élie Ducommun - Élie - Ducommun
        Sesso: m
        Nato/a il 1833-02-19 a Geneva, Switzerland
        Morto/a il 1906-12-07 a Bern, Switzerland
        Wikipedia: https://en.wikipedia.org/wiki/Élie_Ducommun
```

4. Interrogazioni con max, min e avg

In [13]:

```
print("Interrogazioni con max, min e avg")
from math import trunc


# Interrogazione e stampa dei risultati
```

```
risultati = list(collection_nobelPrices2.aggregate([{ "$group": {
        "_id": 'null',
        "max_data": { "$max": "$dateAwarded" } ,
        "min_data": { "$min": "$dateAwarded" } ,
        "max_prize": { "$max": "$prizeAmount" } ,
        "min_prize": { "$min": "$prizeAmount" } ,
        "avg_prize": { "$avg": "$prizeAmount"}
    }}]))[0]
print("I nobel presenti nel database:")
print("\t- sono stati vinti tra il " + risultati['min_data'] + " e il " +
risultati["max_data"])
print("\t- hanno un importo compreso tra " +
get_importo(risultati['min_prize']) + " e " +
get_importo(risultati['max_prize']) + " (corone svedesi), per un importo medio
di: "
    + get_importo(risultati['avg_prize']))
```

```
Interrogazioni con max, min e avg
I nobel presenti nel database:
        - sono stati vinti tra il 1901-12-10 e il 2022-10-07
        - hanno un importo compreso tra 114,935.00 e 10,000,000.00 (corone svedesi), per
un importo medio di: 2,866,560.87
```

5. Quanti Nobel sono stati vinti per ogni categoria?

In [14]:

```
print("Quanti Nobel sono stati vinti per ogni categoria?\n")
print("Categoria\t\t\tNumero di Nobel vinti")
print("-----------------------------------------------------")

category_count = list(collection_nobelPrices2.aggregate([
    { '$match': {"laureates.id" : {'$exists': True}} },     # Nobel con un
vincitore
    { '$unwind': "$category" },
    { '$group': { '_id': "$category" , 'count': { '$sum': 1 } }},
    { '$sort' : {'count':-1} }
]))

for elem in category_count:
    categoria = elem["_id"]
    count = str(elem["count"])
    sep = "\t\t" if len(categoria)>16 else ("\t\t\t\t" if len(categoria)<9
```

```
else "\t\t\t")
    print(categoria +sep+ count)
```

```
Quanti Nobel sono stati vinti per ogni categoria?

Categoria                       Numero di Nobel vinti
-------------------------------------------------------
Physics                         116
Literature                      115
Chemistry                       114
Physiology or Medicine          113
Peace                           103
Economic Sciences               54
```

6. Persone/organizzazioni che hanno vinto piu di un Nobel

In [15]:

```
print("Persone/organizzazioni che hanno vinto piu di un Nobel")

# Richiesta degli input
print("Scegliere un'opzione:\n1. Considerare solo i vincitori che hanno vinto come singoli\n2. Considerare anche coloro che hanno vinto insieme ad altri vincitori")
opzione = ""
while (not(opzione=="1" or opzione=="2")):
    opzione = input(">> ")

stampa_dettagli = chiedi_stampa_dettagli()

# Interrogazione e stampa dei risultati
filtri_ricerca = [
    { '$group': { '_id': "$laureates.id" , 'count': { '$sum': 1 } }},
    { '$match' : {'count': {'$gt':1}}},
    { '$sort' : {'count':-1}},
    { '$lookup': {
            "localField": "_id",
            "from": "laureates2",
            "foreignField": "id",
            "as": "vincitore"
        }
    }
]
if opzione == "2":
    filtri_ricerca = [{ '$unwind': "$laureates" }] + filtri_ricerca
```

```
lista_vincitori = list(collection_nobelPrices2.aggregate(filtri_ricerca))

for elem in lista_vincitori:
    if len(elem["vincitore"]) == 0:
        continue
    print("--------------")
    print("Nobel vinti: " + str(elem["count"]))
    print_vincitore(elem["vincitore"][0], stampa_dettagli)
```

```
Persone/organizzazioni che hanno vinto piu di un Nobel
Scegliere un'opzione:
1. Considerare solo i vincitori che hanno vinto come singoli
2. Considerare anche coloro che hanno vinto insieme ad altri vincitori
>> 2
Stampare anche i dettagli dei vincitori (y/n)?n
--------------
Nobel vinti: 3
Organizzazione: International Committee of the Red Cross
--------------
Nobel vinti: 2
Persona: K. Barry Sharpless
--------------
Nobel vinti: 2
Persona: Frederick Sanger
--------------
Nobel vinti: 2
Organizzazione: Office of the United Nations High Commissioner for Refugees
--------------
Nobel vinti: 2
Persona: Marie Curie
--------------
Nobel vinti: 2
Persona: John Bardeen
--------------
Nobel vinti: 2
Persona: Linus Pauling
```

Interrogazioni con comandi non visti a lezione

1. Stampare i dati di un vincitore random italiano

In [21]:

```
print("Stampare i dati di un vincitore random italiano")

vincitore_italiano = list(collection_laureates2.aggregate([
    {'$match': {"birth.place.countryNow":"Italy"}},     # Cerco tra i vincitori nati in Italia
    { '$sample': { 'size': 1 } }                        # Scelta di un documento in modo random
    ]))
```

```
for v in vincitore_italiano:
    print_vincitore(v, stampa_dettagli=True)
```

```
Stampare i dati di un vincitore random italiano
Persona: Eugenio Montale
        Altri nomi: Eugenio Montale - Eugenio - Montale
        Sesso: m
        Nato/a il 1896-10-12 a Genoa, Italy
        Morto/a il 1981-09-12 a Milan, Italy
        Wikipedia: https://en.wikipedia.org/wiki/Eugenio_Montale
```

2. Stampare gli anni in cui almeno una categoria non e' stata assegnata

In [22]:

```
print("Stampare gli anni in cui almeno una categoria non e' stata assegnata")

lista_nobel = list(collection_nobelPrices2.aggregate([
        { '$match': {"laureates.id" : {'$exists': False}} },            #
Seleziono solo i nobel senza un vincitore
        { '$group': {
            "_id":'$awardYear',                                         #
Raggruppo in base all'anno
            'categorie': {
                '$accumulator': {
# Creo una funzione in JavaScript per concatenare le categorie:
                    'accumulateArgs': ["$category"],
#   - Seleziono il field
                    'init': 'function() { return [] }',
#   - Inizializzo
                    'accumulate': 'function(stringa, s2) { return
stringa.concat(s2) }',    #   - Unisco le due stringhe
                    'merge': 'function(s1, s2) { return s1.concat(s2) }',
#   - Unisco le due stringhe
                    'finalize': 'function(stringa) { return stringa.join(", ")
}',          #   - Metto una virgola tra le categorie
                    'lang': "js"
                }
                }
         }},
         { '$sort': {'_id':1}},                                         #
Stampo in ordine di anno
]))
```

```
for nobel in lista_nobel:
    print("Anno: " + str(nobel["_id"]) + "\tCategorie non assegnate: "+
nobel["categorie"])
```

```
Stampare gli anni in cui almeno una categoria non e' stata assegnata
Anno: 1914      Categorie non assegnate: Literature, Peace
Anno: 1915      Categorie non assegnate: Peace, Physiology or Medicine
Anno: 1916      Categorie non assegnate: Chemistry, Peace, Physics, Physiology or Medicin
e
Anno: 1917      Categorie non assegnate: Chemistry, Physiology or Medicine
Anno: 1918      Categorie non assegnate: Literature, Peace, Physiology or Medicine
Anno: 1919      Categorie non assegnate: Chemistry
Anno: 1921      Categorie non assegnate: Physiology or Medicine
Anno: 1923      Categorie non assegnate: Peace
Anno: 1924      Categorie non assegnate: Chemistry, Peace
Anno: 1925      Categorie non assegnate: Physiology or Medicine
Anno: 1928      Categorie non assegnate: Peace
Anno: 1931      Categorie non assegnate: Physics
Anno: 1932      Categorie non assegnate: Peace
Anno: 1933      Categorie non assegnate: Chemistry
Anno: 1934      Categorie non assegnate: Physics
Anno: 1935      Categorie non assegnate: Literature
Anno: 1939      Categorie non assegnate: Peace
Anno: 1940      Categorie non assegnate: Chemistry, Literature, Peace, Physics, Physiolog
y or Medicine
Anno: 1941      Categorie non assegnate: Chemistry, Literature, Peace, Physics, Physiolog
y or Medicine
Anno: 1942      Categorie non assegnate: Chemistry, Literature, Peace, Physics, Physiolog
y or Medicine
Anno: 1943      Categorie non assegnate: Literature, Peace
Anno: 1948      Categorie non assegnate: Peace
Anno: 1955      Categorie non assegnate: Peace
Anno: 1956      Categorie non assegnate: Peace
Anno: 1966      Categorie non assegnate: Peace
Anno: 1967      Categorie non assegnate: Peace
Anno: 1972      Categorie non assegnate: Peace
```

Scritture e modifiche

In [24]:

```
def chiedi_id_nome_vincitore():
    id = ""
    while id == "":
        name = input("Inserire il nome della persona/organizzazione: ")
        lista_ris = list(collection_laureates2.find({'$or': [{"knownName" :
name}, {"orgName" : name}]} ))

        if len(lista_ris) ==0:
            print("Nessun risultato trovato. Reinserire il nome. ")
            continue

        print("Risultati trovati:")
```

```
        id_list = []
        for vincitore in lista_ris:
            print_vincitore(vincitore, True)
            if "id" in vincitore:
                print("\tid:" + vincitore["id"])
                id_list += [vincitore["id"]]
            print()
        id = input("Inserire il numero di id per confermare (premere invio per
reinserire il nome): ")

        if id not in id_list:
            print("Id inserito non presente nella lista! Reinserire il nome.
")
            id = ""

    vincitore = list(collection_laureates2.find({'id':id}))[0]
    if "orgName" in vincitore:
        id_known_or_org_name = {"id": id, "orgName": vincitore["orgName"]}
    elif "knownName" in vincitore:
        id_known_or_org_name = {"id": id, "knownName": vincitore["knownName"]}
    return id_known_or_org_name
```

1. Inserire un premio nobel ed eventualmente il vincitore

In [26]:

```
print("Inserire dati di un nuovo premio Nobel")
print("I dati non conosciuti possono essere omessi premendo invio")
from datetime import datetime

def check_data(data_str):
    """Questa funzione controlla che la data in input abbia un formato
corretto"""
    if data_str != "":
        datetime.strptime(data_str, "%Y-%m-%d")

def inserisci_nuovo_vincitore():
    """Questa funzione inserisce nella collection un nuovo vincitore"""

    # Richiesta e controlli degli input relativi ai vincitori
    is_persona = input("Il nobel e' stato vinto da una persona (y/n)?") == 'y'
    if is_persona:
```

```
        knownName = input("Nome (principale): ")
        givenName = input("Altro nome: ")
        familyName = input("Nome della famiglia: ")
        fullName = input("Nome completo: ")
        gender = input("Sesso (m/f): ")
        gender = "female" if gender == 'f' else "male"
        birthDate = input("Data di nascita ("+FORMATO_DATA+"): ")
        check_data(birthDate)
        birthPlace_cityNow = input("Citta di nascita: ")
        birthPlace_countryNow = input("Paese: ")

        deathDate = input("Data di morte ("+FORMATO_DATA+"): ")
        check_data(deathDate)
        deathPlace_cityNow = input("Citta di morte: ")
        deathPlace_countryNow = input("Paese: ")
    else:
        orgName = input("Nome (principale): ")
        nativeName = input("Altro nome: ")
        foundedDate = input("Data di fondazione ("+FORMATO_DATA+"): ")
        check_data(foundedDate)
        foundedPlace_cityNow = input("Citta di fondazione: ")
        foundedPlace_countryNow = input("Paese: ")

    wikipedia = input("Link alla pagina wikipedia: ")
    id = str(collection_laureates2.estimated_document_count()+1025)

    # Inserimento del documento e return
    if is_persona:
        collection_laureates2.insert_one({
            "id" : id,
            "knownName": knownName,
            "givenName": givenName,
            "familyName": familyName,
            "fullName": fullName,
            "gender": gender,
            "birth": {
                "date": birthDate,
                "place": {
                    "cityNow": birthPlace_cityNow,
                    "countryNow": birthPlace_countryNow
```

```
                }
            },
            "death": {
                "date": deathDate,
                "place": {
                    "cityNow": deathPlace_cityNow,
                    "countryNow": deathPlace_countryNow
                }
            },
            "wikipedia": wikipedia
        })
        return {"id" : id, "knownName": knownName }
    else:
        collection_laureates2.insert_one({
            "id": id,
            "orgName": orgName,
            "nativeName": nativeName,
            "founded": {
                "date": foundedDate,
                "place": {
                    "cityNow": foundedPlace_cityNow,
                    "countryNow": foundedPlace_countryNow
                }
            },
            "wikipedia": wikipedia
        })
        return  {"id" : id, "orgName": orgName }


# Richiesta dei valori relativi al premio Nobel
FORMATO_DATA = "aaaa-mm-gg"

awardYear = input("Anno: ")
awardYear = int(awardYear) if awardYear != "" else ""

category = input("Categoria: ")

categoryFullName = input("Nome completo della categoria: ")

dateAwarded = input("Data ("+FORMATO_DATA+"): ")
```

```
check_data(dateAwarded)

topMotivation = input("Motivazione: ")

prizeAmount = input("Importo (in corone svedesi): ")
prizeAmount = int(prizeAmount) if prizeAmount != "" else ""



n_laureates = int(input("Numero di vincitori: "))
lista_laureates = []

for i in range(n_laureates):
    # Richiesta dei dati dei vincitori
    print(f"---Dati del vincitore {i+1}---")
    giaPresente = input("Il vincitore e' gia presente (y/n) nel database? ")
== 'y'
    if giaPresente:
        id_known_or_org_name = chiedi_id_nome_vincitore()
    else:
        id_known_or_org_name = inserisci_nuovo_vincitore()

    portion = input("Porzione del Nobel vinta (frazione): ")
    motivation = input("Motivazione: ")

    lista_laureates += [id_known_or_org_name|{"portion":portion,
"motivation":motivation}]

# Inserimento del documento
inserted = collection_nobelPrices2.insert_one({
    "awardYear": awardYear,
    "category": category,
    "categoryFullName": categoryFullName,
    "dateAwarded": dateAwarded,
    "topMotivation": topMotivation,
    "prizeAmount": prizeAmount,
    "laureates": lista_laureates
})

print("\nNobel memorizzato correttamente.")
```

```
Inserire dati di un nuovo premio Nobel
I dati non conosciuti possono essere omessi premendo invio
Anno: 2022
Categoria: Peace
Nome completo della categoria:
Data (aaaa-mm-gg): 2022-10-10
Motivazione: for peace
Importo (in corone svedesi): 10000
Numero di vincitori: 2
---Dati del vincitore 1---
Il vincitore e' gia presente (y/n) nel database? y
Inserire il nome della persona/organizzazione: Max Born
Risultati trovati:
Persona: Max Born
        Altri nomi: Max Born - Max - Born
        Sesso: m
        Nato/a il 1882-12-11 a Wroclaw, Poland
        Morto/a il 1970-01-05 a Göttingen, Germany
        Wikipedia: https://en.wikipedia.org/wiki/Max_Born
        id:61

Inserire il numero di id per confermare (premere invio per reinserire il nome): 61
Porzione del Nobel vinta (frazione): 1/2
Motivazione: for peace 1
---Dati del vincitore 2---
Il vincitore e' gia presente (y/n) nel database? n
Il nobel e' stato vinto da una persona (y/n)?y
Nome (principale): nome cognome
Altro nome: abc
Nome della famiglia: def
Nome completo: ghi
Sesso (m/f): f
Data di nascita (aaaa-mm-gg): 2000-10-10
Citta di nascita: Reggio Emilia
Paese: Italy
Data di morte (aaaa-mm-gg):
Citta di morte:
Paese:
Link alla pagina wikipedia:
Porzione del Nobel vinta (frazione): 1/2
Motivazione: for peace 2

Nobel memorizzato correttamente.
```

2. Modifica dei dati di un vincitore

In [27]:

```python
print("Modifica dei dati di un vincitore")


# Richiesta degli input
id_known_or_org_name = chiedi_id_nome_vincitore()
is_persona = "knownName" in id_known_or_org_name
if is_persona:
    lista_field = [ ("knownName", "Persona"),
            ("fullName", "Altro nome 1:"),
```

```
            ("givenName", "Altro nome 2"),
            ("familyName", "Altro nome 3"),
            ("gender", "Sesso (m/f)"),
            ("birth.date", "Data di nascita"),
            ("birth.place.cityNow", "Citta di nascita"),
            ("birth.place.countryNow", "Paese di nascita"),
            ("death.date", "Data di morte"),
            ("death.place.cityNow", "Citta di morte"),
            ("death.place.countryNow", "Paese di morte"),
            ("wikipedia", "Link di wikipedia")
        ]
else:
    lista_field = [
        ("orgName", "Organizzazione"),
        ("nativeName", "Altro Nome"),
        ("founded.date", "Data di fondazione"),
        ("founded.place.cityNow", "Citta di fondazione"),
        ("founded.place.countryNow", "Paese di fondazione"),
        ("wikipedia", "Link di wikipedia")
    ]

print("Che valore si desidera aggiungere o modificare?")
for i in range(len(lista_field)):
    print(f"{i}. " + lista_field[i][1])
opzione = int(input(">> "))
if opzione < 0 or opzione >= len(lista_field):
    print("Valore non valido!")
    raise MiaStopExecution

nuovo_val = input("Inserire il nuovo valore: ")

if lista_field[opzione][0].__contains__("date"):
    check_data(nuovo_val)
elif lista_field[opzione][0] == "gender":
    gender = "female" if nuovo_val == 'f' else "male"

# Modifico i dati
collection_laureates2.update_one({'id':id_known_or_org_name["id"]},{'$set':
{lista_field[opzione][0]:nuovo_val}})
print("Documento memorizzato correttamente: ")
```

```
print_vincitore(list(collection_laureates2.find({'id':id_known_or_org_name["id"
[0], True)
```

```
Modifica dei dati di un vincitore
Inserire il nome della persona/organizzazione: nome cognome
Risultati trovati:
Persona: nome cognome
        Altri nomi: ghi - abc - def
        Sesso: f
        Nato/a il 2000-10-10 a Reggio Emilia, Italy
        Morto/a il  a ,
        Wikipedia:
        id:2006

Inserire il numero di id per confermare (premere invio per reinserire il nome): 2006
Che valore si desidera aggiungere o modificare?
0. Persona
1. Altro nome 1:
2. Altro nome 2
3. Altro nome 3
4. Sesso (m/f)
5. Data di nascita
6. Citta di nascita
7. Paese di nascita
8. Data di morte
9. Citta di morte
10. Paese di morte
11. Link di wikipedia
>> 8
Inserire il nuovo valore: 2022-10-11
Documento memorizzato correttamente:
Persona: nome cognome
        Altri nomi: ghi - abc - def
        Sesso: f
        Nato/a il 2000-10-10 a Reggio Emilia, Italy
        Morto/a il 2022-10-11 a ,
        Wikipedia:
```