# Assignment 02 Outputs and Code

## Assignment 2.1 Code Output:

Text

Description automatically generated

Measurements.json Output:

{

"(619, 'dyer', 'rad')": {

"visit\_id": 619,

"person\_id": "dyer",

"quantity": "rad",

"reading": 9.82

},

"(619, 'dyer', 'sal')": {

"visit\_id": 619,

"person\_id": "dyer",

"quantity": "sal",

"reading": 0.13

},

"(622, 'dyer', 'rad')": {

"visit\_id": 622,

"person\_id": "dyer",

"quantity": "rad",

"reading": 7.8

},

"(622, 'dyer', 'sal')": {

"visit\_id": 622,

"person\_id": "dyer",

"quantity": "sal",

"reading": 0.09

},

"(734, 'lake', 'sal')": {

"visit\_id": 734,

"person\_id": "lake",

"quantity": "sal",

"reading": 0.05

},

"(734, 'pb', 'rad')": {

"visit\_id": 734,

"person\_id": "pb",

"quantity": "rad",

"reading": 8.41

},

"(734, 'pb', 'temp')": {

"visit\_id": 734,

"person\_id": "pb",

"quantity": "temp",

"reading": -21.5

},

"(735, 'pb', 'rad')": {

"visit\_id": 735,

"person\_id": "pb",

"quantity": "rad",

"reading": 7.22

},

"(735, 'pb', 'sal')": {

"visit\_id": 735,

"person\_id": "pb",

"quantity": "sal",

"reading": 0.06

},

"(735, 'pb', 'temp')": {

"visit\_id": 735,

"person\_id": "pb",

"quantity": "temp",

"reading": -26.0

},

"(751, 'pb', 'rad')": {

"visit\_id": 751,

"person\_id": "pb",

"quantity": "rad",

"reading": 4.35

},

"(751, 'pb', 'temp')": {

"visit\_id": 751,

"person\_id": "pb",

"quantity": "temp",

"reading": -18.5

},

"(752, 'lake', 'rad')": {

"visit\_id": 752,

"person\_id": "lake",

"quantity": "rad",

"reading": 2.19

},

"(752, 'lake', 'sal')": {

"visit\_id": 752,

"person\_id": "lake",

"quantity": "sal",

"reading": 0.09

},

"(752, 'lake', 'temp')": {

"visit\_id": 752,

"person\_id": "lake",

"quantity": "temp",

"reading": -16.0

},

"(752, 'roe', 'sal')": {

"visit\_id": 752,

"person\_id": "roe",

"quantity": "sal",

"reading": 41.6

},

"(837, 'lake', 'rad')": {

"visit\_id": 837,

"person\_id": "lake",

"quantity": "rad",

"reading": 1.46

},

"(837, 'lake', 'sal')": {

"visit\_id": 837,

"person\_id": "lake",

"quantity": "sal",

"reading": 0.21

},

"(837, 'roe', 'sal')": {

"visit\_id": 837,

"person\_id": "roe",

"quantity": "sal",

"reading": 22.5

},

"(844, 'roe', 'rad')": {

"visit\_id": 844,

"person\_id": "roe",

"quantity": "rad",

"reading": 11.25

}

}

People.json Output:

{

"danforth": {

"person\_id": "danforth",

"personal\_name": "Frank",

"family\_name": "Danforth"

},

"dyer": {

"person\_id": "dyer",

"personal\_name": "William",

"family\_name": "Dyer"

},

"lake": {

"person\_id": "lake",

"personal\_name": "Anderson",

"family\_name": "Lake"

},

"pb": {

"person\_id": "pb",

"personal\_name": "Frank",

"family\_name": "Pabodie"

},

"roe": {

"person\_id": "roe",

"personal\_name": "Valentina",

"family\_name": "Roerich"

}

}

Sites.json Output:

{

"DR-1": {

"site\_id": "DR-1",

"latitude": -49.85,

"longitude": -128.57

},

"DR-3": {

"site\_id": "DR-3",

"latitude": -47.15,

"longitude": -126.72

},

"MSK-4": {

"site\_id": "MSK-4",

"latitude": -48.87,

"longitude": -123.4

}

}

Visited.json Output:

{

"(619, 'DR-1')": {

"visit\_id": 619,

"site\_id": "DR-1",

"visit\_date": "1927-02-08"

},

"(622, 'DR-1')": {

"visit\_id": 622,

"site\_id": "DR-1",

"visit\_date": "1927-02-10"

},

"(734, 'DR-3')": {

"visit\_id": 734,

"site\_id": "DR-3",

"visit\_date": "1930-01-07"

},

"(735, 'DR-3')": {

"visit\_id": 735,

"site\_id": "DR-3",

"visit\_date": "1930-01-12"

},

"(751, 'DR-3')": {

"visit\_id": 751,

"site\_id": "DR-3",

"visit\_date": "1930-02-26"

},

"(752, 'DR-3')": {

"visit\_id": 752,

"site\_id": "DR-3",

"visit\_date": NaN

},

"(837, 'MSK-4')": {

"visit\_id": 837,

"site\_id": "MSK-4",

"visit\_date": "1932-01-14"

},

"(844, 'DR-1')": {

"visit\_id": 844,

"site\_id": "DR-1",

"visit\_date": "1932-03-22"

}

}

## Assignment 2.2 Code Output:

Graphical user interface, text, application, email

Description automatically generated

Patient-info.json Output:

{"\_default": {"1": {"person\_id": "danforth", "personal\_name": "Frank", "family\_name": "Danforth", "visits": []}, "2": {"person\_id": "dyer", "personal\_name": "William", "family\_name": "Dyer", "visits": [{"visit\_id": 619, "site\_id": "DR-1", "visit\_date": "1927-02-08", "site": {"site\_id": "DR-1", "latitude": -49.85, "longitude": -128.57}, "measurements": [{"visit\_id": 619, "person\_id": "dyer", "quantity": "rad", "reading": 9.82}, {"visit\_id": 619, "person\_id": "dyer", "quantity": "sal", "reading": 0.13}]}, {"visit\_id": 622, "site\_id": "DR-1", "visit\_date": "1927-02-10", "site": {"site\_id": "DR-1", "latitude": -49.85, "longitude": -128.57}, "measurements": [{"visit\_id": 622, "person\_id": "dyer", "quantity": "rad", "reading": 7.8}, {"visit\_id": 622, "person\_id": "dyer", "quantity": "sal", "reading": 0.09}]}]}, "3": {"person\_id": "lake", "personal\_name": "Anderson", "family\_name": "Lake", "visits": [{"visit\_id": 752, "site\_id": "DR-3", "visit\_date": NaN, "site": {"site\_id": "DR-3", "latitude": -47.15, "longitude": -126.72}, "measurements": [{"visit\_id": 752, "person\_id": "lake", "quantity": "rad", "reading": 2.19}, {"visit\_id": 752, "person\_id": "lake", "quantity": "sal", "reading": 0.09}, {"visit\_id": 752, "person\_id": "lake", "quantity": "temp", "reading": -16.0}]}, {"visit\_id": 837, "site\_id": "MSK-4", "visit\_date": "1932-01-14", "site": {"site\_id": "MSK-4", "latitude": -48.87, "longitude": -123.4}, "measurements": [{"visit\_id": 837, "person\_id": "lake", "quantity": "rad", "reading": 1.46}, {"visit\_id": 837, "person\_id": "lake", "quantity": "sal", "reading": 0.21}]}, {"visit\_id": 734, "site\_id": "DR-3", "visit\_date": "1930-01-07", "site": {"site\_id": "DR-3", "latitude": -47.15, "longitude": -126.72}, "measurements": [{"visit\_id": 734, "person\_id": "lake", "quantity": "sal", "reading": 0.05}]}]}, "4": {"person\_id": "pb", "personal\_name": "Frank", "family\_name": "Pabodie", "visits": [{"visit\_id": 751, "site\_id": "DR-3", "visit\_date": "1930-02-26", "site": {"site\_id": "DR-3", "latitude": -47.15, "longitude": -126.72}, "measurements": [{"visit\_id": 751, "person\_id": "pb", "quantity": "rad", "reading": 4.35}, {"visit\_id": 751, "person\_id": "pb", "quantity": "temp", "reading": -18.5}]}, {"visit\_id": 734, "site\_id": "DR-3", "visit\_date": "1930-01-07", "site": {"site\_id": "DR-3", "latitude": -47.15, "longitude": -126.72}, "measurements": [{"visit\_id": 734, "person\_id": "pb", "quantity": "rad", "reading": 8.41}, {"visit\_id": 734, "person\_id": "pb", "quantity": "temp", "reading": -21.5}]}, {"visit\_id": 735, "site\_id": "DR-3", "visit\_date": "1930-01-12", "site": {"site\_id": "DR-3", "latitude": -47.15, "longitude": -126.72}, "measurements": [{"visit\_id": 735, "person\_id": "pb", "quantity": "rad", "reading": 7.22}, {"visit\_id": 735, "person\_id": "pb", "quantity": "sal", "reading": 0.06}, {"visit\_id": 735, "person\_id": "pb", "quantity": "temp", "reading": -26.0}]}]}, "5": {"person\_id": "roe", "personal\_name": "Valentina", "family\_name": "Roerich", "visits": [{"visit\_id": 752, "site\_id": "DR-3", "visit\_date": NaN, "site": {"site\_id": "DR-3", "latitude": -47.15, "longitude": -126.72}, "measurements": [{"visit\_id": 752, "person\_id": "roe", "quantity": "sal", "reading": 41.6}]}, {"visit\_id": 844, "site\_id": "DR-1", "visit\_date": "1932-03-22", "site": {"site\_id": "DR-1", "latitude": -49.85, "longitude": -128.57}, "measurements": [{"visit\_id": 844, "person\_id": "roe", "quantity": "rad", "reading": 11.25}]}, {"visit\_id": 837, "site\_id": "MSK-4", "visit\_date": "1932-01-14", "site": {"site\_id": "MSK-4", "latitude": -48.87, "longitude": -123.4}, "measurements": [{"visit\_id": 837, "person\_id": "roe", "quantity": "sal", "reading": 22.5}]}]}}}

## Assignment 2.3 Code Output:

Text

Description automatically generated with medium confidence

Graphical user interface, text, application, email

Description automatically generated

## Assignment 2.4 Code Output:

Text, application, email

Description automatically generated

[{"date":"2023-03-09T00:00:00Z","event":"http://www.wikidata.org/entity/Q111458258"},{"date":"2023-03-02T00:00:00Z","event":"http://www.wikidata.org/entity/Q111458314","eventLabel":"2022–23 Biathlon World Cup – Stage 7"},{"date":"2023-03-16T00:00:00Z","event":"http://www.wikidata.org/entity/Q111458340"},{"date":"2023-03-05T00:00:00Z","event":"http://www.wikidata.org/entity/Q111460810","eventLabel":"2023 Vasaloppet"},{"date":"2023-03-12T00:00:00Z","event":"http://www.wikidata.org/entity/Q115801843","eventLabel":"2023 Women's Hockey Junior Africa Cup"},{"date":"2023-03-12T00:00:00Z","event":"http://www.wikidata.org/entity/Q115802035","eventLabel":"2023 Men's Hockey Junior Africa Cup"},{"date":"2023-03-18T00:00:00Z","event":"http://www.wikidata.org/entity/Q115803958","eventLabel":"UFC 286"},{"date":"2023-02-25T00:00:00Z","event":"http://www.wikidata.org/entity/Q115807057","eventLabel":"UFC Fight Night 220"},{"date":"2023-03-04T00:00:00Z","event":"http://www.wikidata.org/entity/Q115857639","eventLabel":"UFC 285"},{"date":"2023-03-11T00:00:00Z","event":"http://www.wikidata.org/entity/Q115857750","eventLabel":"UFC Fight Night 221"}]

## Assignment 2.1 Code (kvdb File Code):

**import** json

**from** pathlib **import** Path

**import** os

**import** pandas **as** pd

**import** s3fs

'''

def read\_cluster\_csv(file\_path, endpoint\_url='https://storage.budsc.midwest-datascience.com'):

s3 = s3fs.S3FileSystem(

anon=True,

client\_kwargs={

'endpoint\_url': endpoint\_url

}

)

return pd.read\_csv(s3.open(file\_path, mode='rb'))

'''

current\_dir **=** Path(os**.**getcwd())**.**absolute()

results\_dir **=** current\_dir**.**joinpath('results')

kv\_data\_dir **=** results\_dir**.**joinpath('kvdb')

kv\_data\_dir**.**mkdir(parents**=True**, exist\_ok**=True**)

people\_json **=** kv\_data\_dir**.**joinpath('people.json')

visited\_json **=** kv\_data\_dir**.**joinpath('visited.json')

sites\_json **=** kv\_data\_dir**.**joinpath('sites.json')

measurements\_json **=** kv\_data\_dir**.**joinpath('measurements.json')

In [2]:

**class** KVDB(object):

**def** \_\_init\_\_(self, db\_path):

self**.**\_db\_path **=** Path(db\_path)

self**.**\_db **=** {}

self**.**\_load\_db()

**def** \_load\_db(self):

**if** self**.**\_db\_path**.**exists():

**with** open(self**.**\_db\_path) **as** f:

self**.**\_db **=** json**.**load(f)

**def** get\_value(self, key):

**return** self**.**\_db**.**get(key)

**def** set\_value(self, key, value):

self**.**\_db[key] **=** value

**def** save(self):

**with** open(self**.**\_db\_path, 'w') **as** f:

json**.**dump(self**.**\_db, f, indent**=**2)

In [3]:

**def** create\_sites\_kvdb():

db **=** KVDB(sites\_json)

df\_sites **=** pd**.**read\_csv('site.csv')

**for** site\_id, group\_df **in** df\_sites**.**groupby('site\_id'):

db**.**set\_value(site\_id, group\_df**.**to\_dict(orient**=**'records')[0])

db**.**save()

**def** create\_people\_kvdb():

db **=** KVDB(people\_json)

df\_people **=** pd**.**read\_csv('person.csv')

**for** person\_id, group\_df **in** df\_people**.**groupby('person\_id'):

db**.**set\_value(person\_id, group\_df**.**to\_dict(orient**=**'records')[0])

db**.**save()

**def** create\_visits\_kvdb():

db **=** KVDB(visited\_json)

df\_visits **=** pd**.**read\_csv('visited.csv')

**for** composite\_id, group\_df **in** df\_visits**.**groupby(["visit\_id", "site\_id"]):

key**=**str(composite\_id)

db**.**set\_value(key, group\_df**.**to\_dict(orient**=**'records')[0])

db**.**save()

**def** create\_measurements\_kvdb():

db **=** KVDB(measurements\_json)

df\_measurements **=** pd**.**read\_csv('measurements.csv')

**for** composite\_id, group\_df **in** df\_measurements**.**groupby(['visit\_id', 'person\_id', 'quantity']):

key**=**str(composite\_id)

db**.**set\_value(key, group\_df**.**to\_dict(orient**=**'records')[0])

db**.**save()

In [4]:

create\_sites\_kvdb()

create\_people\_kvdb()

create\_visits\_kvdb()

create\_measurements\_kvdb()

## Assignment 2.2 Code (documentdb.ipynb):

**from** pathlib **import** Path

**import** json

**import** os

**from** tinydb **import** TinyDB

current\_dir **=** Path(os**.**getcwd())**.**absolute()

results\_dir **=** current\_dir**.**joinpath('results')

kv\_data\_dir **=** results\_dir**.**joinpath('kvdb')

kv\_data\_dir**.**mkdir(parents**=True**, exist\_ok**=True**)

**def** \_load\_json(json\_path):

**with** open(json\_path) **as** f:

**return** json**.**load(f)

**class** DocumentDB(object):

*## You can use the code from the previous example if you would like*

people\_json **=** kv\_data\_dir**.**joinpath('people.json')

visited\_json **=** kv\_data\_dir**.**joinpath('visited.json')

sites\_json **=** kv\_data\_dir**.**joinpath('sites.json')

measurements\_json **=** kv\_data\_dir**.**joinpath('measurements.json')

*# use with open command for all of the json files*

**with** open(sites\_json) **as** f:

\_sites\_Data **=** json**.**load(f)

**with** open(measurements\_json) **as** f:

\_measurements\_Data **=** json**.**load(f)

**with** open(people\_json) **as** f:

\_people\_Data **=** json**.**load(f)

**with** open(visited\_json) **as** f:

\_visit\_Data **=** json**.**load(f)

**def** \_\_init\_\_(self, db\_path):

self**.**\_db\_path **=** Path(db\_path)

self**.**\_db **=** **None**

self**.**\_load\_db()

**def** \_get\_sites(self, site\_id):

'''

Function: Get site data

arguments: site\_id (str)

returns: site (json)

'''

site **=** self**.**\_sites\_Data[str(site\_id)]

**return** site

**def** \_get\_measurements(self, person\_id):

'''

Function: Get measurements data

arguments: person\_id (str)

returns: measurements (json)

'''

measurements **=** []

*# Use for loop to get measurements data added into array*

**for** measurement **in** self**.**\_measurements\_Data**.**values():

**if** str(measurement['person\_id']) **==** str(person\_id):

measurements**.**extend([measurement])

**return** measurements

**def** \_get\_visits(self, visit\_id):

'''

Function: Get visits and sites data

arguments: visit\_id (str)

returns: visit (array)

'''

visit **=** [visit **for** key, visit **in** self**.**\_visit\_Data**.**items() **if** visit['visit\_id'] **==** visit\_id][0]

site\_id **=** visit['site\_id']

site **=** self**.**\_get\_sites(site\_id)

visit['site'] **=** site

**return** visit

**def** \_load\_db(self):

self**.**\_db **=** TinyDB(self**.**\_db\_path)

people **=** self**.**\_people\_Data**.**items()

**for** person\_id, person\_data **in** people:

measurements **=** self**.**\_get\_measurements(person\_id)

visit\_ids **=** set([measurement['visit\_id'] **for** measurement **in** measurements])

visits **=** []

**for** visit\_id **in** visit\_ids:

visit **=** self**.**\_get\_visits(visit\_id)

visit['measurements'] **=** [measurement **for** measurement **in** measurements **if** visit\_id **==** measurement['visit\_id']]

visits**.**append(visit)

person\_data['visits'] **=** visits

*#print(json.dumps(person\_data, indent = 4))*

self**.**\_db**.**insert(person\_data)

In [2]:

db\_path **=** results\_dir**.**joinpath('patient-info.json')

**if** db\_path**.**exists():

os**.**remove(db\_path)

db **=** DocumentDB(db\_path)

## Assignment 2.3 Code (rdbms.ipynb):

**from** pathlib **import** Path

**import** os

**import** sqlite3

**import** s3fs

**import** pandas **as** pd

current\_dir **=** Path(os**.**getcwd())**.**absolute()

results\_dir **=** current\_dir**.**joinpath('results')

kv\_data\_dir **=** results\_dir**.**joinpath('kvdb')

kv\_data\_dir**.**mkdir(parents**=True**, exist\_ok**=True**)

**def** read\_cluster\_csv(file\_path, endpoint\_url**=**'https://storage.budsc.midwest-datascience.com'):

s3 **=** s3fs**.**S3FileSystem(

anon**=True**,

client\_kwargs**=**{

'endpoint\_url': endpoint\_url

}

)

**return** pd**.**read\_csv(s3**.**open(file\_path, mode**=**'rb'))

## Create and Load Measurements Table

In [2]:

**def** create\_measurements\_table(conn):

sql **=** """

CREATE TABLE IF NOT EXISTS measurements (

visit\_id integer NOT NULL,

person\_id text NOT NULL,

quantity text,

reading real,

FOREIGN KEY (visit\_id) REFERENCES visits (visit\_id),

FOREIGN KEY (person\_id) REFERENCES people (people\_id)

);

"""

c **=** conn**.**cursor()

c**.**execute(sql)

**def** load\_measurements\_table(conn):

create\_measurements\_table(conn)

df **=** pd**.**read\_csv('measurements.csv')

measurements **=** df**.**values

c **=** conn**.**cursor()

c**.**execute('DELETE FROM measurements;') *# Delete data if exists*

c**.**executemany('INSERT INTO measurements VALUES (?,?,?,?)', measurements)

## Create and Load People Table

In [3]:

**def** create\_people\_table(conn):

sql **=** """

CREATE TABLE IF NOT EXISTS people (

person\_id text PRIMARY KEY,

personal\_name text NOT NULL,

family\_name text NOT NULL

);

"""

c **=** conn**.**cursor()

c**.**execute(sql)

**def** load\_people\_table(conn):

create\_people\_table(conn)

df **=** pd**.**read\_csv('person.csv')

people **=** df**.**values

c **=** conn**.**cursor()

c**.**execute('DELETE FROM people;') *# Delete data if exists*

c**.**executemany('INSERT INTO people VALUES (?,?,?)', people)

## Create and Load Sites Table

In [4]:

**def** create\_sites\_table(conn):

sql **=** """

CREATE TABLE IF NOT EXISTS sites (

site\_id text PRIMARY KEY,

latitude double NOT NULL,

longitude double NOT NULL

);

"""

c **=** conn**.**cursor()

c**.**execute(sql)

**def** load\_sites\_table(conn):

create\_sites\_table(conn)

df **=** pd**.**read\_csv('site.csv')

sites **=** df**.**values

c **=** conn**.**cursor()

c**.**execute('DELETE FROM sites;') *# Delete data if exists*

c**.**executemany('INSERT INTO sites VALUES (?,?,?)', sites)

## Create and Load Visits Table

In [5]:

**def** create\_visits\_table(conn):

sql **=** """

CREATE TABLE IF NOT EXISTS visits (

visit\_id integer PRIMARY KEY,

site\_id text NOT NULL,

visit\_date text,

FOREIGN KEY (site\_id) REFERENCES sites (site\_id)

);

"""

c **=** conn**.**cursor()

c**.**execute(sql)

**def** load\_visits\_table(conn):

create\_visits\_table(conn)

df **=** pd**.**read\_csv('visited.csv')

visits **=** df**.**values

c **=** conn**.**cursor()

c**.**execute('DELETE FROM visits;') *# Delete data if exists*

c**.**executemany('INSERT INTO visits VALUES (?,?,?)', visits)

## Create DB and Load Tables

In [6]:

db\_path **=** results\_dir**.**joinpath('patient-info.db')

conn **=** sqlite3**.**connect(str(db\_path))

*# TODO: Uncomment once functions completed*

load\_people\_table(conn)

load\_sites\_table(conn)

load\_visits\_table(conn)

load\_measurements\_table(conn)

conn**.**commit()

conn**.**close()

## Assignment 2.4 Code:

[Wikidata Query Service](https://query.wikidata.org/#%23Recent%20Events%0A%20SELECT%20%3Fdate%20%3Fevent%20%3FeventLabel%0A%20WHERE%0A%20%7B%0A%20%20%20%20%23%20find%20events%0A%20%20%20%20%3Fevent%20wdt%3AP31%2Fwdt%3AP279%2a%20wd%3AQ1190554.%0A%20%20%20%20%23%20with%20a%20point%20in%20time%20or%20start%20date%0A%20%20%20%20OPTIONAL%20%7B%20%3Fevent%20wdt%3AP585%20%3Fdate.%20%7D%0A%20%20%20%20OPTIONAL%20%7B%20%3Fevent%20wdt%3AP580%20%3Fdate.%20%7D%0A%20%20%20%20%23%20but%20at%20least%20one%20of%20those%0A%20%20%20%20FILTER%28BOUND%28%3Fdate%29%20%26%26%20DATATYPE%28%3Fdate%29%20%3D%20xsd%3AdateTime%29.%0A%20%20%20%20%23%20not%20in%20the%20future%2C%20and%20not%20more%20than%2031%20days%20ago%0A%20%20%20%20BIND%28NOW%28%29%20-%20%3Fdate%20AS%20%3Fdistance%29.%0A%20%20%20%20FILTER%280%20%3C%3D%20%3Fdistance%20%26%26%20%3Fdistance%20%3C%2031%29.%0A%20%20%20%20%23%20and%20get%20a%20label%20as%20well%0A%20%20%20%20OPTIONAL%20%7B%0A%20%20%20%20%20%20%20%20%3Fevent%20rdfs%3Alabel%20%3FeventLabel.%0A%20%20%20%20%20%20%20%20FILTER%28LANG%28%3FeventLabel%29%20%3D%20%22en%22%29.%0A%20%20%20%20%7D%0A%20%7D%0A%23%20limit%20to%2010%20results%20so%20we%20don%27t%20timeout%0A%20LIMIT%2010) used to generate the .json file.

Graphical user interface, application

Description automatically generated

#Recent Events

SELECT ?date ?event ?eventLabel

WHERE

{

# find events

?event wdt:P31/wdt:P279\* wd:Q1190554.

# with a point in time or start date

OPTIONAL { ?event wdt:P585 ?date. }

OPTIONAL { ?event wdt:P580 ?date. }

# but at least one of those

FILTER(BOUND(?date) && DATATYPE(?date) = xsd:dateTime).

# not in the future, and not more than 31 days ago

BIND(NOW() - ?date AS ?distance).

FILTER(0 <= ?distance && ?distance < 31).

# and get a label as well

OPTIONAL {

?event rdfs:label ?eventLabel.

FILTER(LANG(?eventLabel) = "en").

}

}

# limit to 10 results so we don't timeout

LIMIT 10