Please indicate below the tool you have analyzed for Homework 3, and past the code you have realized.

Your code must include all queries and indexes (if any), and possibly the script you have used to populate the database (in case you used a tool for this, e.g., compass for MongDB, please specify). If you have interacted with the database system through an external programming language, e.g., Python, insert your functions/program.

======================================================================

**Group number: 30**

**Students: Bellini Giacomo 1970896, Pierini Elisa 1859043**

— **Migration from MySQL to MongoDB using Python script**

import mysql.connector

from pymongo.mongo\_client import MongoClient

import urllib

from tqdm import tqdm

for month in tqdm(["january", "february", "march", "april", "may", "june"]):

mydb = mysql.connector.connect(

host="localhost",

user="root",

password="root",

database="cyclist"

)

mycursor = mydb.cursor(dictionary=True)

query = "SELECT \* from " + month + ";"

mycursor.execute(query)

myresult = mycursor.fetchall()

mycursor.close()

uri = "mongodb://localhost:27017"

# Create a new client and connect to the server

myclient = MongoClient(uri)

mydb = myclient["DM"]

mycol = mydb[month]

if len(myresult) > 0:

x = mycol.insert\_many(myresult) # myresult comes from mysql cursor

print(len(x.inserted\_ids))

**— Collections ascending indexes**

**DM.february:** { \_id, ended\_at, member\_casual, start\_station\_id, start\_station\_name, started\_at }

**DM.january:** {\_id, member\_casual, start\_station\_name }

**DM.april:** { \_id, end\_station\_id, start\_station\_name }

**— DISCLAIMER:**

we chose mongodb because the structure of our mysql tables lent itself very well to a document-based database. In some, we were forced to separate aggregations using secondary collections. The reason was the impossibility of obtaining the correct results in a sufficiently short time. The MongoDB compass program could not process the aggregations in time, so it returned an error (**exceeded execution limit error**). Rather than increasing the execution time limit of the aggregations, we decided to opt for splitting the aggregations, thus saving a considerable amount of execution time. Below are screenshots of the error on query\_3 and the subsequent result using the increase in execution time.

Immagine che contiene testo, schermata, software, Icona del computer

Descrizione generata automaticamente

Immagine che contiene testo, schermata, software, Icona del computer

Descrizione generata automaticamente

As can be seen, **200000ms** is too much for our purposes, so the decision to use secondary collections proved to be more effective for the presentation of the work performed.

**— Queries exported in shell language**

**-- 1) SELECT ALL CLASSIC BIKE RIDES THAT HAVE A START AND END STATION ON JANUARY**

**– DM.january**

[

{

$match:

//SELECT ALL CLASSIC BIKE RIDES THAT HAVE

//A START AND END STATION ON JANUARY

{

rideable\_type: /classic\_bike/,

start\_station\_name: {

$ne: null,

},

end\_station\_name: {

$ne: null,

},

},

},

{

$project:

/\*\*

\* specifications: The fields to

\* include or exclude.

\*/

{

ride\_id: 1,

start\_station\_name: 1,

end\_station\_name: 1,

\_id: 0,

},

},

]

**-- 2) COUNT NUMBER OF RIDES GROUP BY "rideable\_type" IN JUNE AND JANUARY, ORDERED BY MONTH**

**– DM.january**

[

{

$group:

/\*\*

\* \_id: The id of the group.

\* fieldN: The first field name.

\*/

//COUNT NUMBER OF RIDES GROUP BY "rideable\_type" IN JUNE

//AND JANUARY, ORDERED BY MONTH

{

\_id: "$rideable\_type",

number\_of\_rides: {

$sum: 1,

},

month: {

$first: "january",

},

},

},

{

$unionWith:

/\*\*

\* coll: The collection name.

\* pipeline: The pipeline on the other collection.

\*/

{

coll: "june",

pipeline: [

{

$group: {

\_id: "$rideable\_type",

number\_of\_rides: {

$sum: 1,

},

month: {

$first: "june",

},

},

},

],

},

},

]

**-- 3) LET'S SEE THE TOTAL NUMBERS OF "casual" AND "member" IN THE START STATION IN JANUARY:**

**– DM.january**

[

{

$lookup:

/\*\*

\* from: The target collection.

\* localField: The local join field.

\* foreignField: The target join field.

\* as: The name for the results.

\* pipeline: Optional pipeline to run on the foreign collection.

\* let: Optional variables to use in the pipeline field stages.

\*/

//LET'S SEE THE TOTAL NUMBERS OF "casual" AND "member"

//IN THE START STATION IN JANUARY:

{

from: "february",

localField: "start\_station\_name",

foreignField: "start\_station\_name",

as: "february\_data",

},

},

{

$out:

/\*\*

\* Provide the name of the output collection.

\*/

"query\_3\_join",

},

]

**– DM.query\_3\_join**

[

{

$unwind:

/\*\*

\* path: Path to the array field.

\* includeArrayIndex: Optional name for index.

\* preserveNullAndEmptyArrays: Optional

\* toggle to unwind null and empty values.

\*/

"$february\_data",

},

{

$group:

/\*\*

\* \_id: The id of the group.

\* fieldN: The first field name.

\*/

{

\_id: "$start\_station\_name",

number\_of\_memberships\_jan: {

$sum: {

$cond: {

if: {

$regexMatch: {

input: "$member\_casual",

regex: "member",

},

},

then: 1,

else: 0,

},

},

},

number\_of\_casuals\_jan: {

$sum: {

$cond: {

if: {

$regexMatch: {

input: "$member\_casual",

regex: "casual",

},

},

then: 1,

else: 0,

},

},

},

number\_of\_memberships\_feb: {

$sum: {

$cond: {

if: {

$regexMatch: {

input:

"$february\_data.member\_casual",

regex: "member",

},

},

then: 1,

else: 0,

},

},

},

number\_of\_casuals\_feb: {

$sum: {

$cond: {

if: {

$regexMatch: {

input:

"$february\_data.member\_casual",

regex: "casual",

},

},

then: 1,

else: 0,

},

},

},

},

},

{

$sort:

/\*\*

\* Provide any number of field/order pairs.

\*/

{

\_id: 1,

},

},

]

**-- 4) FIND THE MOST FAMOUS RIDES ACROSS ALL 6 MONTHS**

**– DM.january**

[

{

$group:

/\*\*

\* \_id: The id of the group.

\* fieldN: The first field name.

\*/

//FIND THE MOST

//FAMOUS RIDES ACROSS ALL 6 MONTHS

{

\_id: {

start\_station\_name:

"$start\_station\_name",

end\_station\_name: "$end\_station\_name",

},

ride\_count: {

$sum: 1,

},

},

},

{

$project:

/\*\*

\* specifications: The fields to

\* include or exclude.

\*/

{

\_id: 0,

start\_station\_name:

"$\_id.start\_station\_name",

end\_station\_name: "$\_id.end\_station\_name",

ride\_count: "$ride\_count",

},

},

{

$unionWith:

/\*\*

\* coll: The collection name.

\* pipeline: The pipeline on the other collection.

\*/

{

coll: "february",

pipeline: [

{

$group: {

\_id: {

start\_station\_name:

"$start\_station\_name",

end\_station\_name:

"$end\_station\_name",

},

ride\_count: {

$sum: 1,

},

},

},

{

$project: {

\_id: 0,

start\_station\_name:

"$\_id.start\_station\_name",

end\_station\_name:

"$\_id.end\_station\_name",

ride\_count: "$ride\_count",

},

},

],

},

},

{

$unionWith:

/\*\*

\* coll: The collection name.

\* pipeline: The pipeline on the other collection.

\*/

{

coll: "march",

pipeline: [

{

$group: {

\_id: {

start\_station\_name:

"$start\_station\_name",

end\_station\_name:

"$end\_station\_name",

},

ride\_count: {

$sum: 1,

},

},

},

{

$project: {

\_id: 0,

start\_station\_name:

"$\_id.start\_station\_name",

end\_station\_name:

"$\_id.end\_station\_name",

ride\_count: "$ride\_count",

},

},

],

},

},

{

$unionWith:

/\*\*

\* coll: The collection name.

\* pipeline: The pipeline on the other collection.

\*/

{

coll: "april",

pipeline: [

{

$group: {

\_id: {

start\_station\_name:

"$start\_station\_name",

end\_station\_name:

"$end\_station\_name",

},

ride\_count: {

$sum: 1,

},

},

},

{

$project: {

\_id: 0,

start\_station\_name:

"$\_id.start\_station\_name",

end\_station\_name:

"$\_id.end\_station\_name",

ride\_count: "$ride\_count",

},

},

],

},

},

{

$unionWith:

/\*\*

\* coll: The collection name.

\* pipeline: The pipeline on the other collection.

\*/

{

coll: "may",

pipeline: [

{

$group: {

\_id: {

start\_station\_name:

"$start\_station\_name",

end\_station\_name:

"$end\_station\_name",

},

ride\_count: {

$sum: 1,

},

},

},

{

$project: {

\_id: 0,

start\_station\_name:

"$\_id.start\_station\_name",

end\_station\_name:

"$\_id.end\_station\_name",

ride\_count: "$ride\_count",

},

},

],

},

},

{

$unionWith:

/\*\*

\* coll: The collection name.

\* pipeline: The pipeline on the other collection.

\*/

{

coll: "june",

pipeline: [

{

$group: {

\_id: {

start\_station\_name:

"$start\_station\_name",

end\_station\_name:

"$end\_station\_name",

},

ride\_count: {

$sum: 1,

},

},

},

{

$project: {

\_id: 0,

start\_station\_name:

"$\_id.start\_station\_name",

end\_station\_name:

"$\_id.end\_station\_name",

ride\_count: "$ride\_count",

},

},

],

},

},

{

$match:

/\*\*

\* query: The query in MQL.

\*/

{

start\_station\_name: {

$ne: null,

},

end\_station\_name: {

$ne: null,

},

},

},

{

$group:

/\*\*

\* \_id: The id of the group.

\* fieldN: The first field name.

\*/

{

\_id: {

start\_station\_name:

"$start\_station\_name",

end\_station\_name: "$end\_station\_name",

},

ride\_count: {

$sum: "$ride\_count",

},

},

},

{

$project:

/\*\*

\* specifications: The fields to

\* include or exclude.

\*/

{

\_id: 0,

start\_station\_name:

"$\_id.start\_station\_name",

end\_station\_name: "$\_id.end\_station\_name",

ride\_count: "$ride\_count",

},

},

{

$sort:

/\*\*

\* Provide any number of field/order pairs.

\*/

{

ride\_count: -1,

},

},

]

**-- 5) Find the busiest time of day for each start station**

**– DM.june**

[

{

$match:

/\*\*

\* query: The query in MQL.

\*/

//Find the busiest time of day for each start station

{

start\_station\_name: {

$ne: null,

},

},

},

{

$group:

/\*\*

\* \_id: The id of the group.

\* fieldN: The first field name.

\*/

{

\_id: {

start\_station\_name:

"$start\_station\_name",

hour: {

$dateToString: {

format: "%H:%m:%S",

date: "$started\_at",

},

},

},

ride\_count: {

$sum: 1,

},

},

},

{

$sort:

/\*\*

\* Provide any number of field/order pairs.

\*/

{

ride\_count: -1,

},

},

]

**-- 6) Find the longest distance across all months**

**– DM.january**

[

{

$group:

/\*\*

\* \_id: The id of the group.

\* fieldN: The first field name.

\*/

//Find the longest distance across all months

{

\_id: "$ride\_id",

max\_distance: {

$max: {

$sqrt: {

$add: [

{

$pow: [

{

$subtract: [

"$start\_lat",

"$end\_lat",

],

},

2,

],

},

{

$pow: [

{

$subtract: [

"$start\_lng",

"$end\_lng",

],

},

2,

],

},

],

},

},

},

},

},

{

$match:

/\*\*

\* query: The query in MQL.

\*/

{

max\_distance: {

$gt: 0,

},

},

},

{

$project:

/\*\*

\* specifications: The fields to

\* include or exclude.

\*/

{

\_id: 0,

ride\_id: "$\_id",

max\_distance: 1,

},

},

{

$sort:

/\*\*

\* Provide any number of field/order pairs.

\*/

{

max\_distance: -1,

},

},

{

$limit:

/\*\*

\* Provide the number of documents to limit.

\*/

100,

},

{

$unionWith:

/\*\*

\* coll: The collection name.

\* pipeline: The pipeline on the other collection.

\*/

{

coll: "february",

pipeline: [

{

$group: {

\_id: "$ride\_id",

max\_distance: {

$max: {

$sqrt: {

$add: [

{

$pow: [

{

$subtract: [

"$start\_lat",

"$end\_lat",

],

},

2,

],

},

{

$pow: [

{

$subtract: [

"$start\_lng",

"$end\_lng",

],

},

2,

],

},

],

},

},

},

},

},

{

$match: {

max\_distance: {

$gt: 0,

},

},

},

{

$project: {

\_id: 0,

ride\_id: "$\_id",

max\_distance: 1,

},

},

{

$sort: {

max\_distance: -1,

},

},

{

$limit: 100,

},

],

},

},

{

$unionWith:

/\*\*

\* coll: The collection name.

\* pipeline: The pipeline on the other collection.

\*/

{

coll: "march",

pipeline: [

{

$group: {

\_id: "$ride\_id",

max\_distance: {

$max: {

$sqrt: {

$add: [

{

$pow: [

{

$subtract: [

"$start\_lat",

"$end\_lat",

],

},

2,

],

},

{

$pow: [

{

$subtract: [

"$start\_lng",

"$end\_lng",

],

},

2,

],

},

],

},

},

},

},

},

{

$match: {

max\_distance: {

$gt: 0,

},

},

},

{

$project: {

\_id: 0,

ride\_id: "$\_id",

max\_distance: 1,

},

},

{

$sort: {

max\_distance: -1,

},

},

{

$limit: 100,

},

],

},

},

{

$unionWith:

/\*\*

\* coll: The collection name.

\* pipeline: The pipeline on the other collection.

\*/

{

coll: "april",

pipeline: [

{

$group: {

\_id: "$ride\_id",

max\_distance: {

$max: {

$sqrt: {

$add: [

{

$pow: [

{

$subtract: [

"$start\_lat",

"$end\_lat",

],

},

2,

],

},

{

$pow: [

{

$subtract: [

"$start\_lng",

"$end\_lng",

],

},

2,

],

},

],

},

},

},

},

},

{

$match: {

max\_distance: {

$gt: 0,

},

},

},

{

$project: {

\_id: 0,

ride\_id: "$\_id",

max\_distance: 1,

},

},

{

$sort: {

max\_distance: -1,

},

},

{

$limit: 100,

},

],

},

},

{

$unionWith:

/\*\*

\* coll: The collection name.

\* pipeline: The pipeline on the other collection.

\*/

{

coll: "may",

pipeline: [

{

$group: {

\_id: "$ride\_id",

max\_distance: {

$max: {

$sqrt: {

$add: [

{

$pow: [

{

$subtract: [

"$start\_lat",

"$end\_lat",

],

},

2,

],

},

{

$pow: [

{

$subtract: [

"$start\_lng",

"$end\_lng",

],

},

2,

],

},

],

},

},

},

},

},

{

$match: {

max\_distance: {

$gt: 0,

},

},

},

{

$project: {

\_id: 0,

ride\_id: "$\_id",

max\_distance: 1,

},

},

{

$sort: {

max\_distance: -1,

},

},

{

$limit: 100,

},

],

},

},

{

$unionWith:

/\*\*

\* coll: The collection name.

\* pipeline: The pipeline on the other collection.

\*/

{

coll: "june",

pipeline: [

{

$group: {

\_id: "$ride\_id",

max\_distance: {

$max: {

$sqrt: {

$add: [

{

$pow: [

{

$subtract: [

"$start\_lat",

"$end\_lat",

],

},

2,

],

},

{

$pow: [

{

$subtract: [

"$start\_lng",

"$end\_lng",

],

},

2,

],

},

],

},

},

},

},

},

{

$match: {

max\_distance: {

$gt: 0,

},

},

},

{

$project: {

\_id: 0,

ride\_id: "$\_id",

max\_distance: 1,

},

},

{

$sort: {

max\_distance: -1,

},

},

{

$limit: 100,

},

],

},

},

]

**-- 7) Find the number of rides per station in february and april**

**– DM.february**

[

{

$lookup:

/\*\*

\* from: The target collection.

\* localField: The local join field.

\* foreignField: The target join field.

\* as: The name for the results.

\* pipeline: Optional pipeline to run on the foreign collection.

\* let: Optional variables to use in the pipeline field stages.

\*/

//Find the number of rides per station in february and april

{

from: "april",

localField: "start\_station\_id",

foreignField: "start\_station\_id",

as: "april",

},

},

{

$project:

/\*\*

\* specifications: The fields to

\* include or exclude.

\*/

{

\_id: 1,

start\_station\_name: 1,

"april.start\_station\_name": 1,

},

},

{

$out:

/\*\*

\* Provide the name of the output collection.

\*/

"query\_7\_join",

},

]

**– DM.query\_7\_join**

[

{

$unwind:

/\*\*

\* path: Path to the array field.

\* includeArrayIndex: Optional name for index.

\* preserveNullAndEmptyArrays: Optional

\* toggle to unwind null and empty values.

\*/

"$april",

},

{

$match:

/\*\*

\* query: The query in MQL.

\*/

{

start\_station\_name: {

$ne: null,

},

},

},

{

$group:

/\*\*

\* \_id: The id of the group.

\* fieldN: The first field name.

\*/

{

\_id: "$start\_station\_name",

num\_rides: {

$sum: 1,

},

},

},

{

$sort:

/\*\*

\* Provide any number of field/order pairs.

\*/

{

num\_rides: -1,

},

},

{

$limit:

/\*\*

\* Provide the number of documents to limit.

\*/

10,

},

]

**-- 8) Find the number of rides that started and ended on each day in February**

**– DM.february repeat two times exchanging $ifNull: ["$started\_at", "$ended\_at" ] with**

**$ifNull: ["$ended\_at" ,"$started\_at"] and out the two aggregations on two different collections: ride\_dates\_start / ride\_dates\_end**

[

{

$facet:

//Find the number of rides

//that started and ended on

//each day in February

{

ride\_dates: [

{

$project: {

ride\_id: 1,

ride\_date: {

$ifNull: [

"$started\_at",

"$ended\_at",

],

},

},

},

{

$match: {

ride\_date: {

$ne: null,

},

},

},

],

start\_rides: [

{

$match: {

started\_at: {

$ne: null,

},

},

},

{

$group: {

\_id: {

ride\_id: "$ride\_id",

started\_at: "$started\_at",

},

},

},

],

end\_rides: [

{

$match: {

ended\_at: {

$ne: null,

},

},

},

{

$group: {

\_id: {

ride\_id: "$ride\_id",

ended\_at: "$ended\_at",

},

},

},

],

},

},

{

$unwind: "$ride\_dates",

},

{

$lookup: {

from: "february",

let: {

ride\_id: "$ride\_dates.ride\_id",

ride\_date: "$ride\_dates.ride\_date",

},

pipeline: [

{

$match: {

$expr: {

$and: [

{

$eq: ["$$ride\_id", "$ride\_id"],

},

{

$eq: [

"$$ride\_date",

"$started\_at",

],

},

],

},

},

},

{

$group: {

\_id: "$ride\_id",

},

},

],

as: "start\_rides",

},

},

{

$lookup: {

from: "february",

let: {

ride\_id: "$ride\_dates.ride\_id",

ride\_date: "$ride\_dates.ride\_date",

},

pipeline: [

{

$match: {

$expr: {

$and: [

{

$eq: ["$$ride\_id", "$ride\_id"],

},

{

$eq: [

"$$ride\_date",

"$ended\_at",

],

},

],

},

},

},

{

$group: {

\_id: "$ride\_id",

},

},

{

$group: {

\_id: null,

ride\_ids: {

$push: "$\_id",

},

},

},

{

$project: {

\_id: 0,

ride\_ids: 1,

},

},

],

as: "end\_rides",

},

},

{

$project: {

ride\_day: {

$dateToString: {

format: "%Y-%m-%d",

date: "$ride\_dates.ride\_date",

},

},

started\_rides: {

$size: "$start\_rides",

},

ended\_rides: {

$size: "$end\_rides",

},

},

},

{

$group: {

\_id: "$ride\_day",

started\_rides: {

$sum: "$started\_rides",

},

ended\_rides: {

$sum: "$ended\_rides",

},

},

},

{

$sort:

/\*\*

\* Provide any number of field/order pairs.

\*/

{

\_id: 1,

},

},

]

**– DM.ride\_dates\_end**

[

{

$lookup: {

from: "ride\_dates\_start",

localField: "\_id",

foreignField: "\_id",

as: "start\_rides",

},

},

{

$unwind: "$start\_rides",

},

{

$project: {

\_id: 1,

started\_rides: "$start\_rides.started\_rides",

ended\_rides: "$ended\_rides",

},

},

]

**-- 9) Count the number of rides by day of the week and member type on June**

**– DM.june**

[

{

$group:

/\*\*

\* \_id: The id of the group.

\* fieldN: The first field name.

\*/

//Count the number of rides by day

//of the week and member type on June

{

\_id: {

member\_casual: "$member\_casual",

day\_of\_week: {

$dayOfWeek: "$started\_at",

},

},

num\_rides: {

$sum: 1,

},

},

},

{

$project:

/\*\*

\* specifications: The fields to

\* include or exclude.

\*/

{

\_id: 0,

member\_casual: "$\_id.member\_casual",

day\_of\_week: {

$arrayElemAt: [

[

"Sunday",

"Monday",

"Tuesday",

"Wednesday",

"Thursday",

"Friday",

"Saturday",

],

{

$subtract: ["$\_id.day\_of\_week", 1],

},

],

},

num\_rides: 1,

},

},

{

$sort:

/\*\*

\* Provide any number of field/order pairs.

\*/

{

num\_rides: -1,

},

},

]

**-- 10) Find the minimum acceptable distance for each ride in February**

**– DM.february**

[

{

$group:

/\*\*

\* \_id: The id of the group.

\* fieldN: The first field name.

\*/

//Find the minimum acceptable

//distance for each ride in February

{

\_id: "$ride\_id",

min\_distance: {

$min: {

$subtract: ["$end\_lng", "$start\_lng"],

},

},

},

},

{

$project:

/\*\*

\* specifications: The fields to

\* include or exclude.

\*/

{

ride\_id: "$\_id.ride\_id",

min\_distance: {

$round: ["$min\_distance", 2],

},

},

},

{

$match:

/\*\*

\* query: The query in MQL.

\*/

{

min\_distance: {

$gt: 0,

},

},

},

{

$sort:

/\*\*

\* Provide any number of field/order pairs.

\*/

{

min\_distance: -1,

},

},

]