Συστήματα Διαχείρισης Δεδομένων Μεγάλου Όγκου

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ Η/Υ ΚΑΙ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ

PROJECT ΕΑΡΙΝΟΥ ΕΞΑΜΗΝΟΥ 2023-24
Έκδοση εγγράφου 1.1 22/04/2024

ΥΠΕΥΘΥΝΟΙ ΚΑΘΗΓΗΤΕΣ: Μεγαλοοικονόμου Βασίλειος, Κομνηνός Ανδρέας

Big Data Applications

Σκοπός εργασίας

Η παρούσα εργασία αποσκοπεί στην εξοικείωση των φοιτητών με τις τρέχουσες τεχνολογίες αποθήκευσης, ανάκτησης και ανάλυσης των Big Data. Τα Big Data εμφανίζονται με μια πληθώρα από μορφές όπως τα web logs, τα internet clickstreams, τα αδόμητα ή ημιδομημένα δεδομένα. Η ανάλυση των Big Data χρησιμοποιεί πηγές δεδομένων οι οποίες παρέμεναν ανεκμετάλλευτες από τις συμβατικές λύσεις. Ζητήματα όπως η διαχείριση ετερογενών κι ανομοιόμορφων σημαντικά μεγάλων δεδομένων από σχεσιακές βάσεις δεδομένων, η ανάκτηση μεγάλων data sets που είναι διάσπαρτα σε ομογενή ή ετερογενή συστήματα, η επεξεργασία φυσικής γλώσσας, η μηχανική μάθηση και τεχνητή νοημοσύνη καθώς και η πρόβλεψη που στηρίζεται σε αδόμητα δεδομένα, ικανοποιούνται πλέον από τα συστήματα ανάλυσης Big Data. Ειδικότερα στην παρούσα εργασία θα ασχοληθούμε και θα δούμε στην πράξη (hands-on) την διαχείριση δεδομένων με την χρήση εργαλείων ανοικτού κώδικα.

Ομάδες

Η εργασία μπορεί να εκπονηθεί από μεμονωμένα άτομα ή ομάδες το πολύ δύο ατόμων.

Προετοιμασία & Documentation

Προαπαιτούμενο SW

Τα ερωτήματα μπορούν να υλοποιηθούν τοπικά στον υπολογιστή σας. Για την υλοποίηση του project στον τοπικό υπολογιστή θα χρειαστείτε τα παρακάτω βοηθητικά εργαλεία:

- Δημιουργία μιας εικονικής μηχανής με Linux (π.χ. ubuntu) – προαιρετικό αλλά συστήνεται ανεπιφύλακτα.
- Εγκατάσταση Java 8 (11 με μικρές αλλαγές στο config)
- Εγκατάσταση Python 3.8+
- Εγκατάσταση Jupyter Notebooks (προαιρετικό αλλά συνιστάται)
- Εγκατάσταση Kafka, Spark, MongoDB και συναφών εργαλείων.

Περιγραφή datasets και εργασιών

Δεδομένα

Για την εργασία θα αξιοποιηθεί ο εξομοιωτής κυκλοφορίας οχημάτων UXSIM. Ο εξομοιωτής είναι γραμμένος σε Python και μπορείτε να δείτε τον κώδικά του στο GitHub.

Μαζί με την εργασία, σας δίνεται πρότυπη υλοποίηση σε Jupyter Notebook ώστε να μπορείτε να δημιουργήσετε μια προσομοίωση διάρκειας μίας ώρας ενός απλού σεναρίου.

Στόχοι εργασίας

Για την εργασία θα υλοποιήσετε ένα τυπικό αγωγό (pipeline) εισόδου, επεξεργασίας σε πραγματικό χρόνο και αποθήκευσης σε NoSQL βάση δεδομένων (MongoDB).

1. Παραγωγή δεδομένων. Με βάση τα αποτελέσματα του εξομοιωτή uxsim θα γράψετε ένα python script που θα στέλνει δεδομένα σε ένα Kafka broker σε τακτά χρονικά διαστήματα.
2. Επεξεργασία σε πραγματικό χρόνο. Τα εισρέοντα δεδομένα από τον Kafka broker θα διαβάζονται από μια υλοποίηση Apache Spark, η οποία εκτελεί real-time επεξεργασία στα δεδομένα.
3. Αποθήκευση σε NoSQL βάση δεδομένων. Τα ωμά δεδομένα και η επεξεργασμένη (από το Spark) μορφή τους αποθηκεύονται σε μια υλοποίηση MongoDB.

Ερώτημα 1: Παραγωγή δεδομένων

1. Εγκαταστήστε και τρέξτε τον εξομοιωτή UXSIM ώστε να παράγετε δεδομένα για το project.

   Ο εξομοιωτής παράγει δεδομένα κίνησης για κάθε όχημα κάθε N δευτερόλεπτα (παράμετρος `deltan` στο ακόλουθο screenshot):

   ```python
   from uxsim import *
   import itertools

   seed = None

   W = World(
       name="",
       deltan=5,
       tmax=3600, #1 hour simulation
       print_mode=1, save_mode=0, show_mode=1,
       random_seed=seed,
       duo_update_time=600
   )
   random.seed(seed)
   ```

   Με τη συνάρτηση `vehicles_to_pandas()` τα δεδομένα παράγονται με τη μορφή pandas dataframe που μπορείτε στη συνέχεια να σώσετε σε csv αρχείο, ή να διαβάσετε απευθείας. Τα πεδία είναι:

   - `name`: κωδικός του οχήματος (ή της ομάδας / διμοιρίας οχημάτων)
   - dn: το μέγεθος της ομάδας (διμοιρίας)
   - `orig`: αφετηρία οχήματος
   - `dest`: προορισμός οχήματος
   - `t`: χρόνος από την αρχή της εκτέλεσης της εξομοίωσης (σε δευτερόλεπτα)
   - `link`: τρέχουσα οδική ακμή που βρίσκεται το όχημα ή ο κωδικός κατάστασης
   - `x`: θέση εντός της οδικής ακμής από την αρχή της (σε μέτρα)
   - `s`: την απόσταση του οχήματος από το προπορευόμενο (σε μέτρα)
   - `v`: τρέχουσα ταχύτητα του οχήματος (χλμ/ώρα)

2. Υλοποιήστε μια υπηρεσία kafka broker ακολουθώντας τις σχετικές οδηγίες εγκατάστασης.
3. Δημιουργήστε ένα νέο topic με όνομα “`vehicle_positions`” μέσω command line (step 3 του οδηγού) ή μέσω της βιβλιοθήκης kafka-python. Στο topic αυτό θα στέλνονται τα δεδομένα που δημιουργήθηκαν από τον εξομοιωτή.

4. Γράψτε ένα script σε python ώστε κάθε *N* δευτερόλεπτα να αποστέλλει τα δεδομένα όλων των οχημάτων που βρίσκονται σε κίνηση προς τον kafka broker του ερωτήματος 2 (δείτε παρακάτω). Για το σκοπό αυτό χρησιμοποιήστε τη βιβλιοθήκη kafka-python.

   Για την αποστολή, δημιουργήστε ένα αντικείμενο της κλάσης `KafkaProducer` και στέλνετε περιοδικά (κάθε *N* δευτερόλεπτα) όλα τα σχετικά δεδομένα. Η θέση κάθε οχήματος αποτελεί ξεχωριστό JSON αντικείμενο που στέλνεται αυτόνομα και ανεξάρτητα από όλες τις άλλες θέσεις. Προσέξτε επίσης ότι θα πρέπει να προσθέσετε τη χρονική σφραγίδα αποστολής των δεδομένων με βάση την ημερομηνία και ώρα που τρέξατε την προσομοίωση. Συνεπώς, για παράδειγμα ας θεωρήσουμε ότι ο producer άρχισε την αποστολή στις `22/4/2024 15:53:45` και ότι τα δεδομένα που έφτιαξε ο simulator απεικονίζονται από το παρακάτω screenshot:

[7]:

	name	dn	orig	dest	t	link	x	s	v
0	0	5	N1	S1	15	N1I1	0.0	125.0	30.0
1	0	5	N1	S1	20	N1I1	100.0	175.0	20.0
2	0	5	N1	S1	25	N1I1	250.0	175.0	30.0
3	0	5	N1	S1	30	N1I1	400.0	-1.0	30.0
4	0	5	N1	S1	35	I1S1	50.0	-1.0	20.0
5	0	5	N1	S1	40	I1S1	200.0	-1.0	30.0
6	0	5	N1	S1	45	I1S1	350.0	-1.0	30.0
7	0	5	N1	S1	45	trip_end	-1.0	-1.0	-1.0
8	1	5	N1	S1	25	waiting_at_origin_node	-1.0	-1.0	-1.0
9	1	5	N1	S1	30	waiting_at_origin_node	-1.0	-1.0	-1.0
10	1	5	N1	S1	35	waiting_at_origin_node	-1.0	-1.0	-1.0
11	1	5	N1	S1	40	N1I1	0.0	175.0	30.0
12	1	5	N1	S1	45	N1I1	150.0	175.0	30.0
13	1	5	N1	S1	50	N1I1	300.0	175.0	30.0

   Έτσι, το 1ο μήνυμα που θα φύγει προς τον kafka broker θα είναι ένα JSON αντικείμενο με βάση το `index 0` του dataset, έχοντας την ακόλουθη δομή:


```
{
    "name":"0",
    "origin":"N1",
    "destination":"S1",
    "time":"22/04/2024 15:54:00",
    "link": "N1I1",
    "position": 0.0,
    "spacing": 125.0,
    "speed": 30.0
}
```


   Προσέξτε ότι η ώρα είναι η ώρα εκκίνησης του producer T + 15 δευτερόλεπτα, με βάση την τιμή που υπάρχει στο πεδίο t του dataset. Αντίστοιχα, για το χρόνο T + 25 δευτερόλεπτα, θα πρέπει να σταλεί μήνυμα για το όχημα 0 αλλά και για το όχημα 1 που φαίνεται για πρώτη φορά στο dataset (`index 8`). Εδώ όμως βλέπουμε ότι το όχημα δεν έχει ξεκινήσει ακόμα (waiting at origin node) οπότε δε θα στείλουμε κάτι για αυτό. Το 1ο μήνυμα για το όχημα 1 θα φύγει στο χρόνο T + 40 δευτερόλεπτα.

5. Υλοποιήστε ένα απλό καταναλωτή (`KafkaConsumer`) για την υποδοχή των μηνυμάτων μέσα από τον Kafka broker ώστε να ελέγξετε ότι η διαδικασία αποστολής και κατανάλωσης των δεδομένων από το Ερώτημα 1 γίνεται σωστά.

Χρήσιμοι σύνδεσμοι:

- Εξομοιωτής UXSIM (documentation και github)
- Βιβλιοθήκη KafkaPython
- Οδηγίες εγκατάστασης Kafka
- Απλή υλοποίηση Kafka producers & consumers σε Python

Ερώτημα 2: Κατανάλωση και επεξεργασία με Spark

1. Εγκαταστήστε το Spark και την απαραίτητη βιβλιοθήκη PySpark σε Python.
2. Στη συνέχεια υλοποιήστε μια διεργασία σε Spark η οποία επεξεργάζεται τα εισερχόμενα δεδομένα και παράγει ένα Spark Dataframe με τα ακόλουθα πεδία:
    - `time`: η χρονική στιγμή από την αρχή της εξομοίωσης
    - `link`: το όνομα μιας ακμής
    - `vcount`: το πλήθος οχημάτων μέσα στην ακμή
    - `vspeed`: η μέση ταχύτητα των οχημάτων μέσα στην ακμή

Χρήσιμοι σύνδεσμοι

- Εγκατάσταση PySpark
- Κατανάλωση δεδομένων από Kafka και επεξεργασία JSON σε Spark
- Επεξεργασία JSON σε Spark

Ερώτημα 3: Αποθήκευση σε MongoDB

1. Εγκαταστήστε την MongoDB.
2. Κατεβάστε τον MongoDB Spark driver και ξεκινήστε το PySpark shell ώστε να τον ενσωματώνει.
3. Σε κατάλληλες συλλογές στη MongoDB, αποθηκεύστε τόσο τα ωμά δεδομένα όπως έρχονται στο Spark από τον Kafka, όσο και τα αποτελέσματα της επεξεργασίας. Για το σκοπό αυτό χρησιμοποιήστε τα foreach / foreachBatch output sinks του Spark.

Υλοποιήστε στη συνέχεια ένα ξεχωριστό python script με το οποίο κάνουμε query την MongoDB και απαντούμε στα ακόλουθα ερωτήματα:

1. Ποια ακμή είχε το μικρότερο πλήθος οχημάτων μεταξύ μιας προκαθορισμένης χρονικής περιόδου;
2. Ποια ακμή είχε τη μεγαλύτερη μέση ταχύτητα μεταξύ μιας προκαθορισμένης χρονικής περιόδου;
3. Ποια ήταν η μεγαλύτερη διαδρομή σε μια προκαθορισμένη χρονική περίοδο;

Χρήσιμοι σύνδεσμοι

- Εγκατάσταση MongoDB (τοπικά ή σε δωρεάν hosted account – μέχρι 512Mb χώρος)
- Εγκατάσταση Mongo Spark Driver (τελευταία έκδοση) [και για Python]
- Ανάγνωση και εξαγωγή δεδομένων μεταξύ Spark και MongoDB
- Έτοιμο docker container με Spark, Mongo & Jupyter [κι εδώ]

Παραδοτέα

Για την εργασία θα πρέπει να παραδώσετε:

1) Τον κώδικα που γράψατε για την εκτέλεση όλων των ερωτημάτων **αποκλειστικά σε γλώσσα Python (αρχεία .py)**.
2) Αναφορά στην οποία περιλαμβάνονται
    a. Περιγραφή του data generator script
    b. Περιγραφή της διαδικασίας εγκατάστασης όλου του pipeline
    c. Ο σχεδιασμός της ΒΔ.
    d. Ερωτήματα DDL και εισαγωγής δεδομένων, με screenshots από την επιτυχή εκτέλεση των τελευταίων.
    e. Ερωτήματα DML για την ανάκτηση δεδομένων και παραγόμενα αποτελέσματα από την εκτέλεση ερωτημάτων.

Να χρησιμοποιηθεί αποκλειστικά το template υποβολής που επισυνάπτεται στην εκφώνηση.

Καταληκτική ημερομηνία υποβολής: 10/6/2024

Επικοινωνία

Για την επιτυχία σας στο project θα χρειαστείτε καθοδήγηση καθώς κι απαντήσεις σε ερωτήματα που ίσως δεν έχουν καλυφθεί στο παρόν κείμενο. Για απορίες μπορείτε να αποστέλλετε στο akomninos@ceid.upatras.gr

Παράρτημα 1 – Πρότυπο αναφοράς άσκησης

Συστήματα Διαχείρισης Δεδομένων Μεγάλου Όγκου

Εργαστηριακή Άσκηση 2023/24

Ονομα	**Επώνυμο**	**ΑΜ**

Βεβαιώνω ότι είμαι συγγραφέας της παρούσας εργασίας και ότι έχω αναφέρει ή παραπέμψει σε αυτήν, ρητά και συγκεκριμένα, όλες τις πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, προτάσεων ή λέξεων, είτε αυτές μεταφέρονται επακριβώς (στο πρωτότυπο ή μεταφρασμένες) είτε παραφρασμένες. Επίσης βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά ειδικά για το συγκεκριμένο μάθημα/σεμινάριο/πρόγραμμα σπουδών.

Έχω ενημερωθεί ότι σύμφωνα με τον εσωτερικό κανονισμό λειτουργίας του Πανεπιστημίου Πατρών άρθρο 50§6, τυχόν προσπάθεια αντιγραφής ή εν γένει φαλκίδευσης της εξεταστικής και εκπαιδευτικής διαδικασίας από οιονδήποτε εξεταζόμενο, πέραν του μηδενισμού, συνιστά βαρύ πειθαρχικό παράπτωμα.

Υπογραφή

___ / ___ / 2024

Υπογραφή

___ / ___ / 2024

Συνημμένα αρχεία κώδικα

Μαζί με την παρούσα αναφορά υποβάλλουμε τα παρακάτω αρχεία κώδικα

Αρχείο	Αφορά το ερώτημα	Περιγραφή/Σχόλιο
Erotima1.py	*1*	*Περιέχει όλα τα ερωτήματα για το ερ. 1*

Τεχνικά χαρακτηριστικά περιβάλλοντος λειτουργίας

[Τεχνικά χαρακτηριστικά φυσικού Η/Υ που χρησιμοποιήθηκε για την εργασία, αν χρησιμοποιήθηκε hosted υπηρεσία μπορείτε απλά να αναφέρετε αυτό αντί για τον πίνακα]

Χαρακτηριστικό	**Τιμή**
CPU model	*Intel i5-10400F*
CPU clock speed	*2.9GHz*
Physical CPU cores	*6*
Logical CPU cores	*12*
RAM	*16*
Secondary Storage Type	*~~HDD~~/SSD*

Ερώτημα 1: Παραγωγή δεδομένων

[περιγράψτε τη δημιουργία δεδομένων εξομοίωσης και την λογική του script παραγωγής δεδομένων ως kafka producer. Δώστε screenshots από την επιτυχή εγκατάσταση του Kafka, και τη δοκιμή της λειτουργίας του μοντέλου producer-consumer]

Ερώτημα 2: Κατανάλωση και επεξεργασία με Spark

[περιγράψτε τη λειτουργία του script κατανάλωσης και επεξεργασίας δεδομένων από το Spark, με κατάλληλα screenshots που δείχνουν την ορθή λειτουργία της διαδικασίας]

Ερώτημα 3: Αποθήκευση σε MongoDB

[παραθέστε: 1. Το σχεδιασμό της βάσης και τα statements δημιουργίας των συλλογών, 2. Τη λειτουργία του script αποθήκευσης των ωμών δεδομένων και των αποτελεσμάτων επεξεργασίας από το Spark στη MongoDB με screenshots των αποτελεσμάτων, 3. Την εκτέλεση του script επερωτημάτων προς τη MongoDB μαζί με τα queries που σχεδιάσατε, και τα αποτελέσματα της εκτέλεσης αυτών με κατάλληλα screenshots].

Σχολιασμός αποτελεσμάτων

[Συνοψίστε τα αποτελέσματα της εμπειρίας σας από το project.]

Βιβλιογραφία

[πηγές που χρησιμοποιήσατε για την εργασία]