DOI: https://doi.org/10.36719/2663-4619/102/232-238

Rəşad Nurullayev
Azərbaycan Dövlət İqtisad Universiteti
magistrant
rashadnf@gmail.com

MONGODB VERİLƏNLƏR BAZASININ TƏDQİQİ: SƏHİYYƏ MƏLUMATLARININ TƏCRÜBƏDƏ TƏTBİQİ VƏ AŞAĞI İNNOVATİV İDARƏETMƏ PROBLEMLƏRİ İLƏ MÜBARİZƏ

Xülasə

Bu tədqiqat işi MongoDB verilənlər bazasının özündə cəmləşdirdiyi əsas xüsusiyyətləri araşdırır və onun müasir texnologiya dünyasındakı yerini müzakirə edir. O, sənəd yönümlü və ənənəvi verilənlər bazaları arasında əhəmiyyətli müqayisəni vurğulayır ki, bu da onların tətbiq oluna bilməsi üçün zəruridir. Bu hesabatın hərtərəfli araşdırılması nəticəsində verilənlər bazasının potensial mənfi və müsbət tərəfləri də ətraflı təsvir edilmişdir.

Məqalə Azərbaycanda Tibbi Məlumatların İdarə Edilməsi ilə bağlı aktual problemlə məşğul olacaq, Fokus-MongoDB verilənlər bazasını daxil etməklə inteqral həllər təqdim edəcək. Həmçinin, səhiyyədə innovativ idarəetmənin aşağı səviyyələrinin həlli üçün bütöv hesab edilə bilən metodları dəqiq müəyyənləşdirəcək və mürəkkəb prosesi olduqca asanlaşdıracaqdır.

Açar sözlər: *MongoDB, verilənlər bazasının xüsusiyyətləri, sənəd yönümlü verilənlər bazaları, ənənəvi verilənlər bazaları, müqayisə, üstünlüklər, mənfi tərəflər, çeviklik, strukturlaşdırılmamış məlumatlar, populyarlıq, tibbi məlumatların idarə edilməsi, Azərbaycan, səhiyyə, innovativ idarəetmə, həllər*

Rashad Nurullayev
Azerbaijan State University of Economics
master student
rashadnf@gmail.com

MongoDB database research: applying health data in practice and combating challenges in innovative low management

Abstract

This research paper explores the main features of the MongoDB database and discusses its place in today's technology world. It highlights an important comparison between document-oriented and traditional databases, which is necessary for their applicability. The potential negatives and positives of the database are also detailed as a result of this report's thorough investigation.

The article will deal with the actual problem of Medical Information Management in Azerbaijan, and will provide integrated solutions by including the Fokus-MongoDB database. Also, it will define holistic methods for solving low levels of innovative management in health care and greatly simplify the complex process.

Keywords: *MongoDB, database features, document-oriented databases, traditional databases, comparison, upsides, downsides, flexibility, unstructured data, popularity, medical data management, Azerbaijan, healthcare, innovative management, solutions*

Giriş

Tibb sektorunu gün keçdikcə inkişaf edir və tibb təhsilində böyük məlumatlar tez-tez istifadə olunur. Vizual analitika sahəsi məlumatların təhlili, istismar üsulları, insanın idrak qabiliyyətləri, məlumat və biliklərin təqdim edilməsi və vizual nümunələrin müəyyənləşdirilməsini bir sıra tətbiqlər əhatə edir.

2007-ci ildə 10gen adlı bir şirkət tərəfindən platforma kimi xidmət məhsulunun komponenti kimi hazırlanmış MongoDB 20 oktyabr 2017-ci ildə açıq şəkildə satılan şirkət oldu. Layihə, axtarılan keyfiyyətlərinə görə tərtibatçılar arasında böyük populyarlıq qazandı, sonradan "parıldayan zirehdə cəngavər" kimi tanınan və tərtibatçıları cəlb etdi.

İstifadə asanlığı, genişlənmə qabiliyyəti və çevikliyi ilə texnologiya dünyasına nəzərəçarpacaq töhfə verdi. Aparıcı NoSQL verilənlər bazası kimi MongoDB pərakəndə satış və maliyyədən tutmuş texnologiya və səhiyyəyə qədər müxtəlif sahələrdə müxtəlif problemlər üçün güc mərkəzi kimi mövcud olmaq potensialını yüksək performansı nümayiş etdirdi (Bellazzi, 2014: 8-13).

İstifadə rahatlığına və bütün müsbət cəhətlərə baxmayaraq, MongoDB həyata keçirmə strategiyaları, düzgün planlaşdırma və MongoDB icmasının dəstəyi kimi kölgə keyfiyyətlərinizi ortaya çıxara bilər. Qeyri-ənənəvi verilənlər bazasından istifadə anlayışı öz başına bir problemdir: məlumatların ardıcıllığı və sxem dizaynı mürəkkəbliyi bütün mənzərənin mürəkkəb hissələridir, düzgün həll etməli olacaqsınız (Chen, 2015).

MongoDB commands	**PostgreSQL commands**
'use <database_name>'	'CREATE DATABASE <name>;'
'db.createCollection(<name>);'	CREATE TABLE <name;>
'db.<collection>.insertOne()'	'INSERT INTO <table>…;'
'db.<collection>.find()'	'SELECT * FROM <table> …;'

Cədvəl (a). MongoDB və PostgreSQL əmrləri arasındakı fərq

Bu verilənlər bazasının üstünlüklərini və çatışmazlıqlarını öyrəndikcə, onun bəzi əsas anlayışlarını aydınlaşdırmaq vacibdir.

Aşağıdakı siyahı bu maraq dairəsini keçmək üçün səyahətə çıxarkən qarşılaşacağınız şərtlərlə bağlı vacib təfərrüatları nümayiş etdirir (Khan, Yaqoob, Hashem, Inayat, Ali, Alam, 2014: 1-18):

- Sənədlər: Sənəd verilənlər bazasındakı qeydlər
- Replika Dəstləri: Yüksək Əlçatımlılığın Təmin Edilməsi
- Parçalanma: Kütləvi məlumat artımını idarə etmək üçün miqyaslılıq
- İndekslər: Sorğu Sürətinin Təkmilləşdirilməsi
- İndi materialı daha da ətraflı izah etmək üçün hər bir nöqtəni nəzərdən keçirək:

Sənədlər: Verilənlər bazasındakı qeydlər nəzərdə tutulur. MongoDB məlumatları müştərilər və xidmətlər arasında məlumat ötürmək üçün istifadə olunan JSON sənədləri kimi saxlayır. Bu sənədlər yüngül və sürətli olması ilə məşhurdur, bu da məlumatların sürətli və səmərəli əldə edilməsini təmin edir.

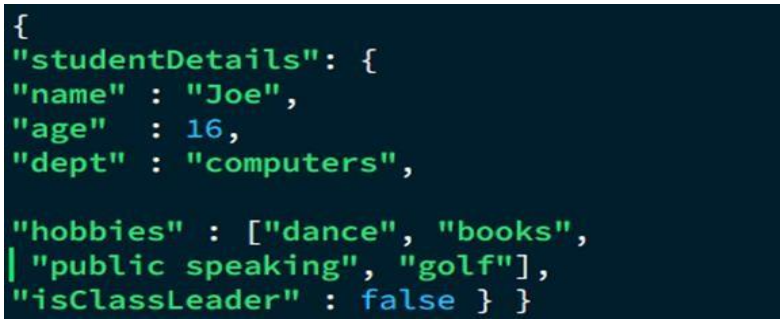
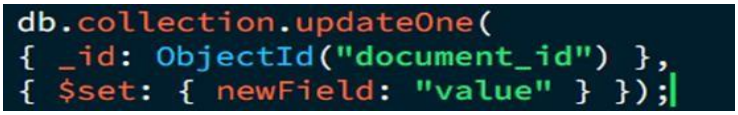


Fig.1 JSON formatında saxlanılan verilənlərin nümunəsi

Daha çox aydınlıq gətirmək üçün yuxarıdakı kod Joe adlı şəxsin çoxlu şəxsi məlumatlarını əks etdirir. Bu, böyük bir verilənlər bazasında daha çox şəxsiyyətlər haqqında bəzi məlumatlar olan daha böyük bir sistemin bir hissəsi olmalı olan məlumat parçasıdır (Rumsfeld, Joynt, Maddox, 2016: 350-359).

Əlaqəli verilənlər bazaları kifayət qədər sərt sxemə malik olsa da və cədvəlin sütunları ilə müəyyən edilsə də, MongoDB istifadəçilərə daha çox azadlıq və daha az vaxt aparan əməliyyatlar təklif edir. Ənənəvi verilənlər bazasına yeni sahə əlavə etməklə siz bütün sxemi dəyişdirmiş olursunuz, bu da məlumatların idarə edilməsinin çevik və rahat yolu deyil. Bunun əksinə olaraq, sənəd yönümlü verilənlər bazamız funksionallığı baxımından sadə olmaqla, sxemsiz bir təbiət nümayiş etdirir.



Şəkil 2. Bu, yeni sahənin necə daxil edilməli olduğunu göstərən koddur

Yuxarıdakı kodda **db.collection.updateOne()** funksiyası müəyyən kolleksiya daxilində sənədi yeniləmək üçün tətbiq edilir.

{id: ObjectId ("document_id")} yenilənəcək sənədi süzgəcdən keçirən və təyin edən meyardır. Bu nümunədə sənəd unikal ObjectId dəyəri ilə istifadə edilən _id sahəsi ilə müəyyən edilir. Nəhayət, bu yeniləmə əməliyyatıdır: **{$set:{ newField: "value"}}.**

Sahənin dəyərini təyin etmək üçün **$set** operatorundan istifadə olunur və bu halda dəyər newField-in "**value**" (dəyərinə) əsasən təyin edilir. MongoDB, əgər mövcud deyilsə, **newField** (yeni Sahə) yaradacaq (Davarzani, Wijk, Maeder, Rickenbacher, Smirnov, Karel, Suter, de Boer, Block, Rolny, Zaugg, Pfisterer, Peeters, Rocca, 2018: 161-173).

Yuxarıda yazılmış koddan istifadə etmək üçün "kolleksiya"nı kolleksiyanızın adı ilə, "document_id"i yeniləmək istədiyiniz sənədin ObjectId ilə, "newField"i əlavə edəcəyiniz sahənin adı ilə əvəz etməlisiniz.

Bütövlükdə, sənəd bazasında qeydlər verilmiş cədvəllə müəyyən edilmir və ciddi struktur tələb etmir, çünki onun istifadəsinin birbaşa olması prosedurun effektiv və tələbkar olmaması üçün kifayətdir.

Replika Dəstləri: Yüksək Əlçatımlılığın Təmin Edilməsi

Növbəti konsepsiyaya keçərkən qeyd etmək lazımdır ki, yüksək əlçatanlığı təmin etmək vacibdir.

Malik olduğunuz məlumatların birdən çox nümunəsini qurmaq və saxlanmas; zamanı MongoDB ilə siz onun dizaynında quraşdırılmış yüksək əlçatanlığa sahibsiniz, çünki MongoDB-də verilənlər bazası yaratdığınız zaman bir xüsusiyyət replika dəsti adlanan məlumatların ən azı iki nüsxəsinin yaradılmasında rol oynayır (Howe, Costanzo, Fey, 2008: 47-50).

Replika dəsti ən azı üç MongoDB dublikatının toplusudur ki, onlar arasında məlumatları təkrarlayır və uğursuzluğun gözlənilə biləcəyi vəziyyətlərdə fasiləyə qarşı təhlükəsizliyi təklif edir. Çox güman ki, niyə ən azı üç kəmiyyət olduğunu düşünməyə başladınız.

Ən azı üç üzv səhvlərə dözümlülük, məlumatların ardıcıllığı və yüksək əlçatanlıq üçün vacibdir. Tək ədəd, qərar qəbul etmə proseduru üçün qovşaqların nasazlığı, şəbəkə bölmələri və s. hallarında belə çoxluğu saxlamaq üçün replika dəsti üçün "mühit" yaradır. MongoDB, cari əsas mövcud olmadıqda, avtomatik uğursuzluq əldə etmək üçün yeni əsas qovşaq seçə bilir. Beləliklə, daimi əməliyyat axını və minimum fasilələrin formalaşmasına zəmanət verilir.

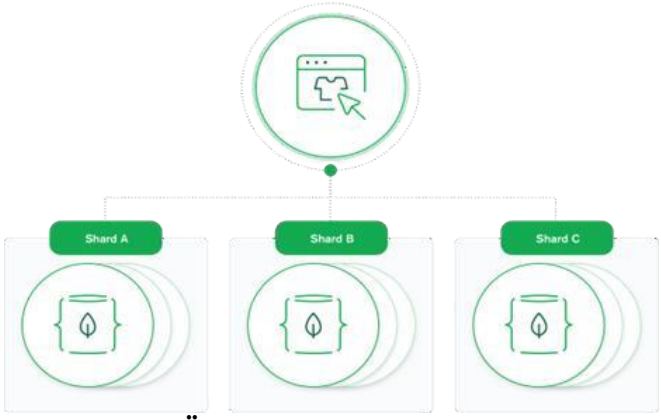
Replika dəstlərinin bütün qeyd olunan aspektlərinə və sistemin işinə əsaslanaraq belə nəticəyə gələ bilərik ki, replika dəstləri MongoDB yerləşdirilməsi üçün etibarlı və möhkəmdir (Duffy, 2016: 494-504).

Parçalanma: Kütləvi məlumat artımını idarə etmək üçün miqyaslılıq.

Parçalanma və onun tətbiqi ilə bağlı bir neçə faktı qeyd etmək də məcburidir və məlumatların birdən çox maşın arasında paylanması üsuluna aiddir. Bu təcrübə paralel emal etməyə imkan verir və adətən tez idarə edilməli olan böyük həcmli müasir məlumat platformaları ilə məşğul olur. Bundan əlavə, sharding miqyaslılığı, eləcə də sürətli funksionallığı ilə sistemlərin işini yaxşılaşdırır.

Bu yaxınlarda Big Data sferasında nəzərəçarpacaq inkişaf, böyük həcmdə məlumatı idarə edə bilən müasir məlumat platformalarına yüksək tələbat yaratdı. Budur, parçalanma kimi müəyyən edilən mühüm texnika əsas rol oynayır. Məlumatların 'üfüqi miqyaslılıq' adlanan çoxsaylı mühərriklər arasında ağıllı və bərabər şəkildə səpilməsi üçün istifadə olunur (Shahzad, 2020).

Parçalanma vasitələrinin özü serverlərin klasterli şəkildə bölüşdürülməsinə əsaslanır, yəni "düyünə" daha çox server əlavə edilə bilər.



Şəkil 3. Üç parçanın birləşməsi fiquru

Məlumat adətən yayılır və kiçik miqyaslı hissələrə bölünür, hər bir parça isə bir alt dəst alır və sorğuları müstəqil şəkildə emal edir. Bu, iş yükünün klasterin bütün hissələrinə necə bərabər bölündüyünü sübut edir. Bütövlükdə, təsvir olunan yanaşma, onlarla effektiv şəkildə məşğul olmaqla, daim artan həcmdə məlumatların artmasına kömək edir.

İndekslər: İndekslər kolleksiyadakı hər bir sənədi, indeksləri oxumaq əvəzinə indeksi skan edən sorğulara görünən təsirinə görə diqqət çəkir.

Düzgün seçilərsə, indekslər sorğuların yerinə yetirilməsini asanlaşdıraraq bu sorğuların sürətinə kömək edə bilər. Ümumiyyətlə iki növ indeks var (Chowdhury, 2019):

Tək sahəli indekslər: Tək sahəli indekslər müəyyən bir sənəddə bir sahə daxilində yaradılır və məlumatların müəyyən bir sahəyə görə çeşidlənməsinə əsaslanan sorğuları sürətləndirmək üçün faydalıdır.

Mürəkkəb indekslər: Bu növ indekslər, əksinə, sorğular üçün faydalıdır sənəd daxilində çoxlu sahələr üzrə filtrləmə və çeşidləmə əsasında.

Nəticə olaraq, indekslər MongoDB verilənlər bazalarının işini optimallaşdırmaq və artırmaqda və sorğuların icra prosesini təkmilləşdirməkdə mühüm rol oynayır. Bütün güclü keyfiyyətlər və

imkanlar verilənlər bazasının malik olduğu bütün potensialın aydın göstəricisi olsa da, MongoDB-nin bütün uyğunluğunu göstərə biləcəyi hər hansı real həyat tətbiqetməsinin mövcud olub-olmadığını düşünə bilərsiniz. Bu baxımdan, biz Azərbaycanda Tibbi Məlumatların İdarəolunması ilə bağlı bir hadisəni müşahidə edəcəyik və əvvəllər qeyd olunan bütün xüsusiyyətləri eyni vaxtda birləşdirərək, onları ardıcıl şəkildə açmağa çalışacağıq (Alharbi, 2018).

Səhiyyə məlumat bazasının idarə edilməsi problemləri və həll məlumatları.

Səhiyyədə innovativ idarəetmənin daha yüksək səviyyədə, vahid səhiyyə informasiya sisteminin olmaması, məlumatların keyfiyyəti və etibarlılığı ilə bağlı problemlə, tibbi texnologiya və xidmətlərin məhdud qiymətləndirilməsi diqqət yetirmək zəriridir. Səhiyyə sahəsində innovativ idarəetmənin aşağı səviyyədə olmasıdır. Bu mövzu səhiyyəni təkmilləşdirmək və tibbi xidmət sahəsində əlçatanlığı artırmaq üçün innovativ texnologiyalarda irəliləyişlərə ehtiyacı artırır.

MongoDB-nin həyati xüsusiyyətlərini keçdiyimiz və onlara toxunduğumuz üçün "təcrübə ən yaxşısını edir" qaydasını nəzərə alaraq praktiki təcrübədən keçməyimiz tələb olunur (Alzahrani, 2017).

Bu vəziyyətdə MongoDB-dən istifadə etmək üçün ilk addım məlumatları CSV, JSON, XML və ya hər hansı digər məqbul məlumat formatına çevirməkdir. Fayl hazır olduqdan sonra məlumat və verilənlər bazası arasında əlaqəni dəstəkləyəcək kodların yazılması əsas təcrübədir. Seçim olaraq qəbul etdiyiniz proqramlaşdırma dilindən asılı olaraq kodunuz bu işdəki bloklardan fərqli olacaq. Bununla belə, əsaslandırma və problem həlli təbiətcə bir-birinə yaxın görünməlidir. Aşağıda ayırd edəcəyiniz kodların siyahısı daha böyük mənzərənin qaçılmaz hissəsidir (Kumar, 2016):

Kod (a):

```
            collection.insertMany(documents);
            System.out.println("Data inserted successfully.");
        } catch (Exception e) {
            System.err.println("Failed to insert data: " + e.getMessage());
        } finally {
            mongoClient.close();
        }}
}import com.mongodb.client.MongoClient;
import com.mongodb.client.MongoClients;
import com.mongodb.client.MongoCollection;
import com.mongodb.client.MongoDatabase;
import org.bson.Document;
import java.nio.file.Files;
import java.nio.file.Paths;

import java.util.ArrayList;
import java.util.List;

new *
public class InsertData {
    new *
    public static void main(String[] args) {
```

```
import com.mongodb.client.MongoClient;
import com.mongodb.client.MongoClients;
import com.mongodb.client.MongoCollection;
import com.mongodb.client.MongoDatabase;
import org.bson.Document;
import java.nio.file.Files;
import java.nio.file.Paths;

import java.util.ArrayList;
import java.util.List;

new *
public class InsertData {
    new *
    public static void main(String[] args) {

        MongoClient mongoClient = MongoClients.create("mongodb://localhost:27017");
        MongoDatabase database = mongoClient.getDatabase( s: "JavaProject");
        MongoCollection<Document> collection = database.getCollection( s: "user_profiles");
        try {
            String json = new String(Files.readAllBytes(Paths.get( first: "something.json")));
            Document document = Document.parse(json);
            List<Document> documents = new ArrayList<>();
            documents.add(document);
            collection.insertMany(documents);
```

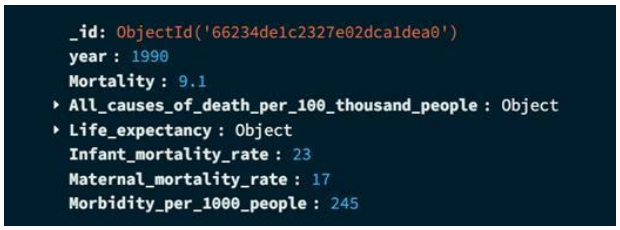
Kod (b).

```
package org.example;

import com.mongodb.client.MongoClients;
import com.mongodb.client.MongoClient;



import com.mongodb.client.MongoDatabase;

no usages  new *
public class MongoDB {
    1 usage
    private static final String DATABASE_NAME = "JavaProject";
    no usages
    private static final String COLLECTION_NAME = "e-commerce";
    1 usage
    private static final String CONNECTION_STRING = "mongodb://localhost:27017";

    no usages  new *
    public static MongoDatabase getDatabase() {
        MongoClient mongoClient = null;
        try {
            mongoClient = MongoClients.create(CONNECTION_STRING);
            return mongoClient.getDatabase(DATABASE_NAME);
        } catch (Exception e) {
```

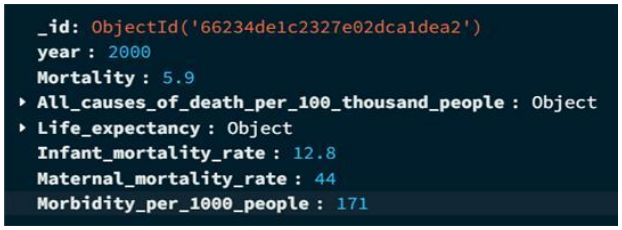
```
        MongoClient mongoClient = MongoClients.create("mongodb://localhost:27017");
        MongoDatabase database = mongoClient.getDatabase( s: "JavaProject");
        MongoCollection<Document> collection = database.getCollection( s: "user_profiles");
        try {
            String json = new String(Files.readAllBytes(Paths.get( first: "something.json")));
            Document document = Document.parse(json);
            List<Document> documents = new ArrayList<>();
            documents.add(document);
            collection.insertMany(documents);
            System.out.println("Data inserted successfully.");
        } catch (Exception e) {
            System.err.println("Failed to insert data: " + e.getMessage());
        } finally {
            mongoClient.close();
        }}
}
```

Bu fraqmentlər MongoDB ilə qarşılıqlı əlaqə üçün istifadə olunan əsas addımlardır. Onlar birlikdə onurğa sütunu kimi xidmət edir və verilmiş Java kodları ilə verilənlər bazası arasında körpü yaradırlar.

Və nəhayət, son addım kompüterinizdə saxlanılan faylı (bu halda JSON) verilənlər bazasına idxal etmək olacaq. Məlumat sorğuya bənzər kod fraqmentlərində saxlanmalıdır:



Ekran görüntüsü 1.



Ekran görüntüsü 2.

Ekran görüntüləri 1 və 2 'JavaProject' adlı layihə çərçivəsində verilənlər bazasında saxlanılan məlumatların vizual təsvirləridir (Zeng, Zhou, 2018).

Nəticə

MongoDB-nin səhiyyə məlumat bazaları üçün nə qədər vacib olduğuna baxdıq. Bu, Azərbaycanda olduğu kimi səhiyyə sistemlərinin üzləşdiyi problemlərin həllinə kömək edir. MongoDB xüsusi xüsusiyyətlərə, üstünlüklərə və istifadələrə malikdir. MongoDB-nin səhiyyə məlumatlarını yaxşı idarə etmək üçün güclü bir yol olduğunu gördük.

MongoDB ilə səhiyyə qrupları məlumatları sənədlərdə saxlaya bilər. MongoDB böyümək asandır və mövcud qalır. Bu, qruplara pis məlumatlar, köhnə idarəetmə üsulları və bölünmüş sistemlər kimi problemləri həll etməyə kömək edir. MongoDB çevik və istifadəsi asandır. Beləliklə, tibb işçiləri asanlıqla və etibarlı şəkildə böyük həcmdə məlumatları saxlaya, əldə edə və öyrənə bilərlər. Nəticədə, bu üsuldan istifadə edilməsi, daha yaxşı xəstə baxımına gətirib çıxarır.

Gələcəkdə MongoDB-ni AI və ML kimi yeni texnologiyalarla birləşdirmək səhiyyənin idarə edilməsini və qərarların qəbulunu daha da yaxşılaşdıra bilər. Səhiyyə təşkilatları MongoDB-dən istifadə etməyə davam etsələr və yeni ideyalar irəli sürsələr, onlar böyük fərq yarada, səhiyyəni daha yaxşı və daha sürətli edə bilərlər. Xülasə etmək üçün, MongoDB səhiyyə mütəxəssislərinin böyük dəyişikliklər etmək və səhiyyə nəticələrini yaxşılaşdırmaq üçün istifadə edə biləcəyi həqiqətən güclü bir vasitədir.

Ədəbiyyat

1. Bellazzi, R. (2014). Big data and biomedical informatics: A challenging opportunity. IMIA Yearbook Med. Informat., vol. 23, № 1, pp.8-13.
2. Chen, H. (2015). Design and Implementation of Medical Image Management System Based on MongoDB. In 2015 IEEE International Conference on Bioinformatics and Biomedicine (BIBM).
3. Khan, N., Yaqoob, I., Hashem, I.A.T., Inayat, Z., Ali, W.K.M., Alam, M. (2014). "Big data: Survey, technologies, opportunities, and challenges", Sci. World J., vol., pp.1-18, Jul.
4. Rumsfeld, J.S., Joynt, K.E., Maddox, T.M. (2016). "Big data analytics to improve cardiovascular care: Promise and challenges". Nature Rev. Cardiol., vol. 13, № 6, pp.350-359, Jun.
5. Davarzani, N., Wijk, S.S.V., Maeder, M.T., Rickenbacher, P., Smirnov, E., Karel, J., Suter, T., de Boer, R.A., Block, D., Rolny, V., Zaugg, C., Pfisterer, M.E., Peeters, R., Rocca, H.P.B.L. (2018). "Novel concept to guide systolic heart failure medication by repeated biomarker testing – Results

from TIME-CHF in context of predictive, preventive, and personalized medicine", EPMA J., vol. 9, № 2, pp.161-173, Jun.
6. Howe, D., Costanzo, M., Fey, P. (2008). Big data: the future of biocuration. Nature, 455, pp.47-50.
7. Duffy, D.J. (2016). Problems, challenges and promises: perspectives on precision medicine. Brief Bioinform, 17, pp.494-504.
8. Shahzad, M. (2020). "Healthcare Big Data Analytics using MongoDB". In 2020 International Conference on Sustainable Technologies for Industry 4.0 (STI).
9. Chowdhury, A.N.M.R. (2019). "Development of a Healthcare Information System with MongoDB", In 2019 IEEE International Conference on Electrical, Computer and Communication Engineering (ECCE).
10. Alharbi, K. (2018). "Integration of MongoDB with IoT and Healthcare Information System". In 2018 International Conference on Advanced Science and Engineering (ICOASE).
11. Alzahrani, A. (2017). "Efficient and Secure Healthcare Data Management System using MongoDB". In 2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC).
12. Kumar, N. (2016). "Secure Electronic Health Record Management System Using MongoDB." In 2016 2nd International Conference on Contemporary Computing and Informatics (IC3I).
13. Zeng, L., Zhou, Z. (2018). Application of MongoDB Database in the Management of Medical Data. In 2018 IEEE International Conference on Smart Cloud (SmartCloud).

Göndərilib: 17.03.2024 Qəbul edilib: 30.04.2024