

بسم الله الرحمن الرحيم

تمرین چهارم پایگاه داده پیشرفته

ریحانه گودرزی

کار با دیتابیس mongo DB

استاد جناب آقای دکتر رشنو

آذر ۱۴۰۳

۱. ورود به `mongodb` ۲. ایجاد پایگاه داده مورد نظر ۳. ایجاد کالکشن (جدول)

```

C:\mongodb>mongo --port 27017 --directConnection=true && serverSelectionTimeoutMS=3000
Microsoft Windows [Version 10.0.19045.5131]
(c) Microsoft Corporation. All rights reserved.

C:\Users\IT\mongodb>
C:\Users\IT\mongodb> mongo --port 27017 --directConnection=true && serverSelectionTimeoutMS=3000
MongoDB shell version: 6.0.1
connecting to: mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=3000&appName=mongosh+2.3.3
Using MongoDB: 6.0.1
Using MongoShell: 2.3.3
MongoDB 6.0.1 is available for download: https://www.mongodb.com/docs/mongodb-6.0/download/shell
For mongosh info see: https://www.mongodb.com/docs/mongosh

-----
The server generated these startup warnings when booting
2024-12-01T13:22:20.528Z:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
-----

> use lorestanUniv
switched to db lorestanUniv
lorestanUniv> db.createCollection("student", {
  "options": {}
})
lorestanUniv> db.createCollection("student", {validator:{$jsonSchema:(bsonType:"object", required:["$STDID", "$FName", "$LName", "$Father", "$Birth", "$BornCity", "$Address", "$PostalCode", "$CPhone", "$Hphone", "$Department", "$Major", "$Married", "$ID"], properties:{STDID:(bsonType:"string", pattern:"^[0-9]{11}$"}, FName:(bsonType:"string", "max length 10", pattern:"^[\u0600-\u06FF]{1,10}$"), LName:(bsonType:"string", "max length 10", pattern:"^[\u0600-\u06FF]{1,10}$"}, Birth:(bsonType:"string", $BornCity:(bsonType:"string", Address:(bsonType:"string", maxlength:100), PostalCode:(bsonType:"string", pattern:"^[0-9]{10}$"), CPhone:(bsonType:"string", pattern:"^09[0-9]{9}$"), Hphone:(bsonType:"string", pattern:"^09[0-9]{9}$"), Department:(bsonType:"string", pattern:"^[0-9]{11}$"}, FName:(bsonType:"string", "max length 10", pattern:"^[\u0600-\u06FF]{1,10}$"), LName:(bsonType:"string", "max length 10", pattern:"^[\u0600-\u06FF]{1,10}$"}, ID:(bsonType:"string", "max length 10", pattern:"^[\u0600-\u06FF]{1,10}$"}, Birth:(bsonType:"string", $BornCity:(bsonType:"string", Address:(bsonType:"string", maxlength:100), PostalCode:(bsonType:"string", pattern:"^[0-9]{10}$"), CPhone:(bsonType:"string", pattern:"^09[0-9]{9}$"), Hphone:(bsonType:"string", pattern:"^09[0-9]{9}$"), Department:(enum:["Engineering", "lecture", "economic"]), Major:(enum:["Computer", "riazi", "English", "Electric"]), Married:(enum:["Single", "Married"])}, ID:(bsonType:"string", pattern:"^[0-9]{10}$"))}}))
Uncaught: \u0600-\u06FF{1,10}$", LName:(bsonType:"string", "max length 10", pattern:"^[\u0600-\u06FF]{1,10}$"),
    }, Hphone:(bsonType:"string", "max length 100", PostalCode:(bsonType:"string", pattern:"^[0-9]{10}$"), CPhone:(bsonType:"string", pattern:"^09[0-9]{9}$"),
    }, Hphone:(bsonType:"string", "max length 100", PostalCode:(bsonType:"string", pattern:"^[0-9]{10}$"), CPhone:(bsonType:"string", pattern:"^09[0-9]{9}$"),
    }, LName:(bsonType:"string", "max length 10", pattern:"^[\u0600-\u06FF]{1,10}$"), Birth:(bsonType:"string", $BornCity:(bsonType:"string", Address:(bsonType:"string", maxlength:100),
    "string", pattern:"^[0-9]{10}$"), CPhone:(bsonType:"string", pattern:"^09[0-9]{9}$"), Hphone:(bsonType:"string", pattern:"^[0-9]{11}$"), Department:(enum:["Engineering", "lecture", "economic"]), Major
    (enum:["Computer", "riazi", "English", "Electric"]), Married:(enum:["Single", "Married"]}, ID:(bsonType:"string", pattern:"^[0-9]{10}$"))));tanliniv

> |
2 |

LorestanUniv>
LorestanUniv>
LorestanUniv> db.createCollection("student", {validator:{$jsonSchema:(bsonType:"object", required:["$STDID", "$FName", "$LName", "$Father", "$Birth", "$BornCity", "$Address", "$PostalCode", "$CPhone", "$Hphone",
"$Department", "$Major", "$Married", "$ID"], properties:{STDID:(bsonType:"string", pattern:"^[0-9]{11}$"}, FName:(bsonType:"string", "max length 10", pattern:"^[\u0600-\u06FF]{1,10}$"), LName:(bsonType:"st
"$D$"), CPhone:(bsonType:"string", pattern:"^09[0-9]{9}$"), Hphone:(bsonType:"string", pattern:"^09[0-9]{9}$"), Department:(enum:["Engineering", "lecture", "economic"]), Major:(enum:["Compu
ter", "riazi", "English", "Electric"]), Married:(enum:["Single", "Married"])}, ID:(bsonType:"string", pattern:"^[0-9]{10}$"))});

```

۴. ایجاد جداول خواسته شده در تمرین

```

mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
lorestan@nivi>
lorestan@nivi>
lorestan@nivi>
lorestan@nivi>
lorestan@nivi>
lorestan@nivi>
lorestan@nivi>
lorestan@nivi>
lorestan@nivi> db.createCollection("student", { validator: { $jsonSchema: { bsonType: "object", required: ["$IID", "$FName", "$LName", "$Father", "$Birth", "$IDS", "$BornCity", "$Address", "$PostalCode", "$CPhone", "$HPHONE", "$Department", "$Major", "$Married", "$ID"], properties: { $IID: { bsonType: "string", pattern: "[0-9]{11}$" }, FName: { bsonType: "string", maxLength: 10, pattern: "[\u0600-\u06FF]{1,10}$" }, LName: { bsonType: "string", maxLength: 10, pattern: "[\u0600-\u06FF]{1,10}$" }, Father: { bsonType: "string", maxLength: 10, pattern: "[\u0600-\u06FF]{1,10}$" }, Birth: { bsonType: "string", IDS: { bsonType: "string", pattern: "[0-9]{6}$" }, BornCity: { bsonType: "string", pattern: "[0-9]{11}$" }, Address: { bsonType: "string", maxLength: 100 }, PostalCode: { bsonType: "string", pattern: "[0-9]{10}$" }, CPhone: { bsonType: "string", pattern: "[0-9]{11}$" }, HPHONE: { bsonType: "string", pattern: "[0-9]{11}$" }, Department: { bsonType: "string", enum: ["Engineering", "Lecture", "Economic"] }, Major: { bsonType: "string", enum: ["Computer", "Mathematics", "English", "Electrical"] }, Married: { bsonType: "string", enum: ["Single", "Married"] }, ID: { bsonType: "string", pattern: "[0-9]{10}$" } } } });
MongoServerError(NamespaceExists): namespace lorestan.nivi.student already exists, but with different options: { uuid: UUID("bce0fcc9-a232-47e0-b992-3508d5dd58a2") }
lorestan@nivi> db.student.drop();
true
lorestan@nivi> db.createCollection("student", { validator: { $jsonSchema: { bsonType: "object", required: ["$IID", "$FName", "$LName", "$Father", "$Birth", "$IDS", "$BornCity", "$Address", "$PostalCode", "$CPhone", "$HPHONE", "$Department", "$Major", "$Married", "$ID"], properties: { $IID: { bsonType: "string", pattern: "[0-9]{11}$" }, FName: { bsonType: "string", maxLength: 10, pattern: "[\u0600-\u06FF]{1,10}$" }, LName: { bsonType: "string", maxLength: 10, pattern: "[\u0600-\u06FF]{1,10}$" }, Father: { bsonType: "string", maxLength: 10, pattern: "[\u0600-\u06FF]{1,10}$" }, Birth: { bsonType: "string", IDS: { bsonType: "string", pattern: "[0-9]{6}$" }, BornCity: { bsonType: "string", pattern: "[0-9]{11}$" }, Address: { bsonType: "string", maxLength: 100 }, PostalCode: { bsonType: "string", pattern: "[0-9]{10}$" }, CPhone: { bsonType: "string", pattern: "[09]{0-9}{11}$" }, HPHONE: { bsonType: "string", pattern: "[0-9]{11}$" }, Department: { bsonType: "string", enum: ["Engineering", "Lecture", "Economic"] }, Major: { bsonType: "string", enum: ["Computer", "Mathematics", "English", "Electrical"] }, Married: { bsonType: "string", enum: ["Single", "Married"] }, ID: { bsonType: "string", pattern: "[0-9]{10}$" } } } });
ok: 1

```


۶. ایجاد کاربران با سطوح دسترسی:

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
lorestanUniv> db.CourseRegister.insertOne({ CID: "10001", CName: "00000 00000000", Department: "Engineering", Credit: 3, SID: "12345678901", FName: "000", LName: "00000" });db.CourseRegister.insertOne({ CID: "10001", CName: "00000 00000000", Department: "Engineering", Credit: 3, SID: "12345678901", FName: "000", LName: "00000" });
Uncaught:
MongoServerError: Document failed validation
Additional information: {
  failingDocumentId: ObjectId('674d08641d7887cb920d8193'),
  details: {
    operatorName: '$jsonSchema',
    schemaRulesNotSatisfied: [
      {
        operatorName: 'properties',
        propertiesNotSatisfied: [
          {
            propertyName: 'CName',
            details: [
              {
                operatorName: 'pattern',
                specifiedAs: { pattern: '^[0-9]+$' },
                reason: 'regular expression did not match',
                consideredValue: '00000 00000000'
              }
            ]
          }
        ]
      }
    ]
  }
}
lorestanUniv>
lorestanUniv>
lorestanUniv>
lorestanUniv>
lorestanUniv>
lorestanUniv>
lorestanUniv>
lorestanUniv>
lorestanUniv>
lorestanUniv>
lorestanUniv> db.createUser({ user: "Saman", pwd: "password123", roles: [ { role: "dbAdmin", db: "lorestanUniv" }, { role: "readWrite", db: "lorestanUniv" } ] });db.createUser({ user: "Armin", pwd: "password123", roles: [ { role: "read", db: "lorestanUniv" } ] });db.createUser({ user: "Sara", pwd: "password123", roles: [ { role: "readWrite", db: "lorestanUniv", collection: "student" }, { role: "read", db: "lorestanUniv" } ] });
```

۷. دستور مشاهده پایگاه داده موجود:

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
> 1 | const session = db.getMongo().startSession();session.startTransaction();try const dbSession = session.getDatabase("lorestanUniv"); course dbSession.course.deleteOne({ CID: 101 }); CourseRegister dbSession.CourseRegister.deleteMany({ CID: 101 });PresentedCourses dbSession.PresentedCourses.deleteMany({ CID: 101 });session.commitTransaction(); } catch (error) {session.abortTransaction(); print("Transaction aborted due to error:", error); } finally {session.endSession(); }
2 |
lorestanUniv>
lorestanUniv> const session = db.getMongo().startSession();session.startTransaction();try const dbSession = session.getDatabase("lorestanUniv"); course dbSession.course.deleteOne({ CID: 101 }); CourseRegister dbSession.CourseRegister.deleteMany({ CID: 101 });PresentedCourses dbSession.PresentedCourses.deleteMany({ CID: 101 });session.commitTransaction(); } catch (error) {session.abortTransaction(); print("Transaction aborted due to error:", error); } finally {session.endSession(); }
Uncaught:
SyntaxError: Unexpected token, expected "(" (1:76)
> 1 | const session = db.getMongo().startSession();session.startTransaction();try const dbSession = session.getDatabase("lorestanUniv"); course dbSession.course.deleteOne({ CID: 101 }); CourseRegister dbSession.CourseRegister.deleteMany({ CID: 101 });PresentedCourses dbSession.PresentedCourses.deleteMany({ CID: 101 });session.commitTransaction(); } catch (error) {session.abortTransaction(); print("Transaction aborted due to error:", error); } finally {session.endSession(); }
2 |
lorestanUniv>
lorestanUniv> show dbs;
lorestanUniv 104.00 KiB
admin         132.00 KiB
config       108.00 KiB
local        40.00 KiB
lorestanUniv> show collections;
course
CourseRegister
lecturer
PresentedCourses
student
lorestanUniv> db.getCollectionInfos({name:"student"});
TypeError: db.getCollectionInfos is not a function
lorestanUniv> db.getCollectionInfos({ name: "student" });
[
  {
    name: 'student',
    type: 'collection',
    options: {
      validator: {
        '$jsonSchema': {
          bsonType: 'object',
          required: [
            'SID',
            'FName',
            'LName',
            'Father',
            'Birth',
            'IDS',
            'BornCity',
            'Address'
          ]
        }
      }
    }
  }
]
```


۸. مشاهده کالکشن های موجود در این پایگاه داده :

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
roles: [
  { role: 'readWrite', db: 'LorestanUniv' },
  { role: 'read', db: 'LorestanUniv' }
],
mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
}
ok: 1
}
LorestanUniv> db.getUser("Saman");
{
  _id: 'LorestanUniv.Saman',
  userId: UUID("75016ee7-cbe4-4b56-9a01-f014a41280e3"),
  user: 'Saman',
  db: 'LorestanUniv',
  roles: [
    { role: 'readWrite', db: 'LorestanUniv' },
    { role: 'dbAdmin', db: 'LorestanUniv' }
  ],
  mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
}
LorestanUniv> db.student.getIndexes();
[ { v: 2, key: { _id: 1 }, name: '_id' } ]
LorestanUniv> show collections;
course
CourseRegister
lecturer
PresentedCourses
student
LorestanUniv>
```

۹. مشاهده اطلاعات کالکشن ها:

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
course
CourseRegister
lecturer
PresentedCourses
student
LorestanUniv> db.getCollectionInfos((name:"student"));
LorestanUniv> db.getCollectionInfos is not a function
LorestanUniv> db.getCollectionInfos({ name: "student" });
[
  {
    name: 'student',
    type: 'collection',
    options: {
      validator: {
        '$jsonSchema': {
          bsonType: 'object',
          required: [
            '_id',
            'FName',
            'LName',
            'Father',
            'Birth',
            'IDS',
            'BornCity',
            'Address',
            'PostalCode',
            'CPhone',
            'HPhone',
            'Department',
            'Major',
            'Married',
            'ID'
          ],
          properties: {
            _id: { bsonType: 'string', pattern: '^[0-9]{11}$' },
            FName: {
              bsonType: 'string',
              maxLength: 10,
              pattern: '^[0-9]{1,10}$'
            },
            LName: {
              bsonType: 'string',
              maxLength: 10,
              pattern: '^[0-9]{1,10}$'
            },
            Father: {
              bsonType: 'string',
              maxLength: 10,
              pattern: '^[0-9]{1,10}$'
            },
            Birth: { bsonType: 'string' },
            IDS: { bsonType: 'string', pattern: '^[0-9]{6}$' },
            BornCity: { bsonType: 'string' },
            Address: { bsonType: 'string', maxLength: 100 },
            PostalCode: { bsonType: 'string', pattern: '^[0-9]{10}$' },
            CPhone: { bsonType: 'string', pattern: '^09[0-9]{9}$' },
            HPhone: { bsonType: 'string', pattern: '^[0-9]{11}$' },
            Department: { bsonType: 'string', pattern: '^[0-9]{11}$' },
            Major: { bsonType: 'string', pattern: '^[0-9]{11}$' },
            Married: { bsonType: 'string', pattern: '^[0-9]{11}$' },
            ID: { bsonType: 'string', pattern: '^[0-9]{11}$' }
          }
        }
      }
    }
  }
]
```

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
CPhone: { bsonType: 'string', pattern: '^09[0-9]{9}$' },
HPhone: { bsonType: 'string', pattern: '^0[0-9]{11}$' },
Department: { bsonType: 'string', enum: [Array] },
Major: { bsonType: 'string', enum: [Array] },
Married: { bsonType: 'string', enum: [Array] },
ID: { bsonType: 'string', pattern: '^0[0-9]{10}$' }
}
},
info: {
  readOnly: false,
  uuid: UUID('9574b282-71af-4be9-810d-34e4be1d6938')
},
idIndex: { v: 2, key: { _id: 1 }, name: '_id_' }
}
}

lorestanUniv> db.getCollectionInfos({ name: "lecturer" });
[
  {
    name: 'lecturer',
    type: 'collection',
    options: {
      validator: {
        '$jsonSchema': {
          bsonType: 'object',
          required: [
            'LID', 'FName',
            'LName', 'ID',
            'Department', 'Major',
            'Birth', 'Residency',
            'Address', 'PostalCode',
            'CPhone', 'HPhone'
          ],
          properties: {
            LID: { bsonType: 'string', pattern: '^0[0-9]{6}$' },
            FName: {
              bsonType: 'string',
              maxLength: 10,
              pattern: '^[a-zA-Z]{1,10}$'
            },
            LName: {
              bsonType: 'string',
              maxLength: 10,
              pattern: '^[a-zA-Z]{1,10}$'
            },
            ID: { bsonType: 'string', pattern: '^0[0-9]{10}$' },
            Department: { bsonType: 'string', enum: [Array] },
            Major: { bsonType: 'string', enum: [Array] },
            Birth: { bsonType: 'string' },

```

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
}
},
info: {
  readOnly: false,
  uuid: UUID('ddb2c710-cb43-46e2-a25d-45564470823c')
},
idIndex: { v: 2, key: { _id: 1 }, name: '_id_' }
}
}

lorestanUniv> db.getCollectionInfos({ name: "course" });
[
  {
    name: 'course',
    type: 'collection',
    options: {
      validator: {
        '$jsonSchema': {
          bsonType: 'object',
          required: [ 'CID', 'CName', 'Department', 'Credit' ],
          properties: {
            CID: { bsonType: 'string', pattern: '^0[0-9]{5}$' },
            CName: { bsonType: 'string', maxLength: 25, pattern: '^[a-zA-Z]{1,25}$' },
            Department: { bsonType: 'string', enum: [Array] },
            Credit: { bsonType: 'int', minimum: 1, maximum: 4 }
          }
        }
      },
      info: {
        readOnly: false,
        uuid: UUID('c1ee57ad-9132-4ffa-97dc-e13738bdad4c')
      },
      idIndex: { v: 2, key: { _id: 1 }, name: '_id_' }
    }
  }
]

lorestanUniv> db.getCollectionInfos({ name: "CourseRegister" });
[
  {
    name: 'CourseRegister',
    type: 'collection',
    options: {
      validator: {
        '$jsonSchema': {
          bsonType: 'object',
          required: [
            'CID',
            'CName',
            'Department',
            'Credit',
            'SIN'
          ],

```

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
info: {
  readOnly: false,
  uuid: UUID("c1ee57ad-9132-4ffa-97dc-e13738bdadc4")
},
idIndex: { v: 2, key: { _id: 1 }, name: '_id_' }
}
LorestanUniv> db.getCollectionInfos({ name: "CourseRegister" });
[
  {
    name: 'CourseRegister',
    type: 'collection',
    options: {
      validator: {
        '$jsonSchema': {
          bsonType: 'object',
          required: [
            'CID',
            'CName',
            'Department',
            'Credit',
            'SID',
            'FName',
            'LName'
          ],
          properties: {
            CID: { bsonType: 'string', pattern: '^[0-9]{5}$' },
            CName: { bsonType: 'string', maxLength: 25, pattern: '^[0-0]+$' },
            Department: { bsonType: 'string', enum: [Array] },
            Credit: { bsonType: 'int', minimum: 1, maximum: 4 },
            SID: { bsonType: 'string', pattern: '^[0-9]{11}$' },
            FName: {
              bsonType: 'string',
              maxLength: 10,
              pattern: '^[0-0]{1,10}$'
            },
            LName: {
              bsonType: 'string',
              maxLength: 10,
              pattern: '^[0-0]{1,10}$'
            }
          }
        }
      },
      info: {
        readOnly: false,
        uuid: UUID("82ecb714-d3c7-4a1b-a38d-879128267850")
      },
      idIndex: { v: 2, key: { _id: 1 }, name: '_id_' }
    }
  }
]
```

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
LorestanUniv> db.getCollectionInfos({ name: "PresentedCourses" });
[
  {
    name: 'PresentedCourses',
    type: 'collection',
    options: {
      validator: {
        '$jsonSchema': {
          bsonType: 'object',
          required: [
            'CID',
            'CName',
            'Department',
            'Credit',
            'LID',
            'FName',
            'LName'
          ],
          properties: {
            CID: { bsonType: 'string', pattern: '^[0-9]{5}$' },
            CName: { bsonType: 'string', maxLength: 25, pattern: '^[0-0]+$' },
            Department: { bsonType: 'string', enum: [Array] },
            Credit: { bsonType: 'int', minimum: 1, maximum: 4 },
            LID: { bsonType: 'string', pattern: '^[0-9]{6}$' },
            FName: {
              bsonType: 'string',
              maxLength: 10,
              pattern: '^[0-0]{1,10}$'
            },
            LName: {
              bsonType: 'string',
              maxLength: 10,
              pattern: '^[0-0]{1,10}$'
            }
          }
        }
      },
      info: {
        readOnly: false,
        uuid: UUID("318defb6-3612-4dc6-9399-54014e63eec4")
      },
      idIndex: { v: 2, key: { _id: 1 }, name: '_id_' }
    }
  }
]
```

۱۰. مشاهده داده های موجود در یک کالکشن:

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
}
LorestanUniv> db.student.getIndexes();
[ { v: 2, key: { _id: 1 }, name: '_id_' } ]
LorestanUniv> show collections;
course
courseRegister
lecturer
PresentedCourses
student
LorestanUniv> db.student.find();
{
  _id: ObjectId('674d064d1d7887cb920d8190'),
  STID: '12345678901',
  FName: 'محمد',
  LName: 'محمدی',
  Fathers: 'محمد',
  Birth: '1375/09/23',
  IDS: '123456',
  BornCity: 'تهران',
  Address: 'تهران ۱۰',
  PostalCode: '1234567890',
  CPhone: '09123456789',
  HPhone: '02112345678',
  Department: 'Engineering',
  Major: 'Computer',
  Married: 'Single',
  ID: '1234567890'
}
LorestanUniv> db.lecturer.find();
{
  _id: ObjectId('674d074a1d7887cb920d8191'),
  LID: '654321',
  FName: 'محمد',
  LName: 'محمدی',
  ID: '0906543210',
  Department: 'Engineering',
  Major: 'Electrical',
  Birth: '1365/09/15',
  BornCity: 'تهران',
  Address: 'تهران 5',
  PostalCode: '1234567890',
  CPhone: '09123456789',
  HPhone: '03112345678'
}
LorestanUniv>
```

۱۱. مشاهده اطلاعات کاربرانی که در MongoDB ایجاد شده‌اند:

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
pattern: '^[0-9]{1,10}$'
}
}
},
info: {
  readOnly: false,
  uuid: UUID('318defb6-3612-4dc6-9399-54014e63eec4')
},
idIndex: { v: 2, key: { _id: 1 }, name: '_id_' }
}
LorestanUniv> db.getUsers();
users: [
  {
    _id: 'LorestanUniv.Armin',
    userId: UUID('655e7345-2d0c-4a96-a8e3-490d94fd2e0c'),
    user: 'Armin',
    db: 'LorestanUniv',
    roles: [ { role: 'read', db: 'LorestanUniv' } ],
    mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
  },
  {
    _id: 'LorestanUniv.Saman',
    userId: UUID('75010ee7-cbe4-4b56-9a01-f014a41280e3'),
    user: 'Saman',
    db: 'LorestanUniv',
    roles: [
      { role: 'readWrite', db: 'LorestanUniv' },
      { role: 'dbAdmin', db: 'LorestanUniv' }
    ],
    mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
  },
  {
    _id: 'LorestanUniv.Sara',
    userId: UUID('41ff16a2-b98f-43b7-b1ea-c856fa07da7d'),
    user: 'Sara',
    db: 'LorestanUniv',
    roles: [
      { role: 'readWrite', db: 'LorestanUniv' },
      { role: 'read', db: 'LorestanUniv' }
    ],
    mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
  }
],
ok: 1
LorestanUniv>
```


۱۲. دستور مشاهده اطلاعات دقیق تر درباره کاربر خاص سامان (ادمین):

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
{
  LorestanUniv> db.getUsers();
{
  users: [
    {
      _id: 'LorestanUniv.Armin',
      userId: UUID('655e7345-2d0c-4a96-a8e3-490d94fd2e0c'),
      user: 'Armin',
      db: 'LorestanUniv',
      roles: [ { role: 'read', db: 'LorestanUniv' } ],
      mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
    },
    {
      _id: 'LorestanUniv.Saman',
      userId: UUID('75016ee7-che4-4b56-9a01-f014a41280e3'),
      user: 'Saman',
      db: 'LorestanUniv',
      roles: [
        { role: 'readWrite', db: 'LorestanUniv' },
        { role: 'dbAdmin', db: 'LorestanUniv' }
      ],
      mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
    },
    {
      _id: 'LorestanUniv.Sara',
      userId: UUID('41ff16a2-b98f-43b7-blea-c856fa07da7d'),
      user: 'Sara',
      db: 'LorestanUniv',
      roles: [
        { role: 'readWrite', db: 'LorestanUniv' },
        { role: 'read', db: 'LorestanUniv' }
      ],
      mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
    }
  ],
  ok: 1
}
LorestanUniv> db.getUser("Saman");
{
  _id: 'LorestanUniv.Saman',
  userId: UUID('75016ee7-che4-4b56-9a01-f014a41280e3'),
  user: 'Saman',
  db: 'LorestanUniv',
  roles: [
    { role: 'readWrite', db: 'LorestanUniv' },
    { role: 'dbAdmin', db: 'LorestanUniv' }
  ],
  mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
}
```

۱۳. مشاهده نمایه‌ها (Indexes) در یک کالکشن خاص مثل student:

```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
{
  users: [
    {
      _id: 'LorestanUniv.Armin',
      userId: UUID('655e7345-2d0c-4a96-a8e3-490d94fd2e0c'),
      user: 'Armin',
      db: 'LorestanUniv',
      roles: [ { role: 'read', db: 'LorestanUniv' } ],
      mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
    },
    {
      _id: 'LorestanUniv.Saman',
      userId: UUID('75016ee7-che4-4b56-9a01-f014a41280e3'),
      user: 'Saman',
      db: 'LorestanUniv',
      roles: [
        { role: 'readWrite', db: 'LorestanUniv' },
        { role: 'dbAdmin', db: 'LorestanUniv' }
      ],
      mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
    },
    {
      _id: 'LorestanUniv.Sara',
      userId: UUID('41ff16a2-b98f-43b7-blea-c856fa07da7d'),
      user: 'Sara',
      db: 'LorestanUniv',
      roles: [
        { role: 'readWrite', db: 'LorestanUniv' },
        { role: 'read', db: 'LorestanUniv' }
      ],
      mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
    }
  ],
  ok: 1
}
LorestanUniv> db.getUser("Saman");
{
  _id: 'LorestanUniv.Saman',
  userId: UUID('75016ee7-che4-4b56-9a01-f014a41280e3'),
  user: 'Saman',
  db: 'LorestanUniv',
  roles: [
    { role: 'readWrite', db: 'LorestanUniv' },
    { role: 'dbAdmin', db: 'LorestanUniv' }
  ],
  mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
}
LorestanUniv> db.student.getIndexes();
[ { v: 2, key: { _id: 1 }, name: '_id' } ]
LorestanUniv>
```

۱۴. استفاده از دستورات commit ، rollback ، abort با ایجاد savepoint :

```

C:\mongodb\mongodb\127.0.0.1:27017\directConnection>true?serverSelectionTimeoutMS=2000
lorestanUniv> try {
...   print("Transaction started.");
...   const course_id = "34567";
...   const delete_Course = db.course.deleteOne({CID: course_id}, {session});
...   if (delete_Course.deletedCount == 0) {
...     throw new Error("No record found in course collection.");
...   }
...   print("Deleted one record from course collection.");
...   const delete_CourseRegister = db.CourseRegister.deleteMany({ CID: course_id }, { session });
...   print("Deleted related records from CourseRegister collection.");
...   const delete_PresentedCourses = db.PresentedCourses.deleteMany({ CID: course_id }, { session });
...   print("Deleted related records from PresentedCourses collection.");
...   session.commitTransaction();
...   print("Transaction committed.");
... } catch (error) {
...   print("ERROR: " + error.message);
...   session.abortTransaction();
...   print("Transaction ABORTED.");
... } finally {
...   session.endSession();
...   print("Session End.");
... }
Transaction started.
ERROR: s7.inTransaction is not a function
Transaction ABORTED.
Session End.

```