Name: Krishna Somani
Roll No. : 2021058

Report - Assignment 1

1. **About RDBMS Schema**

1. Database Structure: There are Six main tables: Departments, Instructors, Students, Courses, Enrollments, and CourseInstructor.

2. Departments Table: Holds unique department_id and department_name.

3. Instructors Table: Stores instructor details and links to Departments via department_id with cascading deletes.

4. Students Table: Records student info linked to Departments through department_id.

5. Courses Table: Represents courses with foreign keys to ensure valid associations with departments and instructors.

6. Enrollments Table:
Captures student-course relationships, including enrollment_date and grade, with foreign key links.

7. CourseInstructor Table:
Manages many-to-many relationships between courses and instructors with foreign key enforcement.

Full schema is attached as a code file.

2. About Migration and MongoDB Schema

Data Structures Created During Migration

1. **Students Collection**:
   - Each document includes fields such as:
     - _id: Unique identifier for the student.
     - first_name and last_name: Names of the student.
     - email: Contact information.
     - birth_date: Date of birth.
     - enrollment_year: Year the student enrolled.
     - department: An embedded document with department details, including:
       - department_id: Identifier for the department.
       - department_name: Name of the department.
     - enrollments: An array of embedded documents representing courses, each with fields like course_id, course_name, credits, and grade.
2. **Courses Collection**:
   - Each document comprises:
     - _id: Unique identifier for the course.
     - course_name: Title of the course.
     - course_code: Code for the course.
     - credits: Credit hours associated with the course.
     - is_core: Boolean indicating if it is a core course.
     - instructor: An embedded document containing:
       - instructor_id: Identifier for the instructor.
       - instructor_name: Name of the instructor.
     - department: An embedded document with:
       - department_id: Identifier for the department.
       - department_name: Name of the department.
3. **Instructors Collection**:
   - Each document consists of:
     - _id: Unique identifier for the instructor.
     - first_name and last_name: Instructor's names.
     - email: Contact information.
     - hire_date: Date of hiring.
     - department: An embedded document containing:
       - department_id: Identifier for the department.
       - department_name: Name of the department.

More on data migration:

Steps involved:

1. Data Extraction: Data was retrieved from PostgreSQL using SQL queries, including students, enrollments, courses, and instructors, with necessary joins to gather related information.

2. Data Cleaning: Dates were converted to Python *datetime* objects.Null values in critical fields were addressed to prevent inconsistencies.

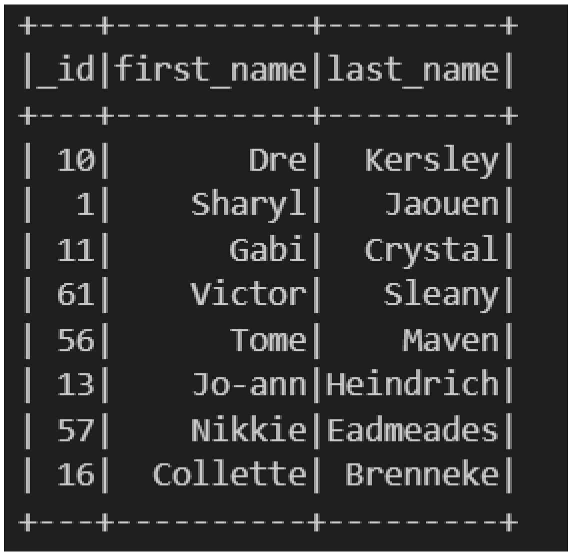
3. Data Transformation: Enrollments were embedded within student documents as arrays to maintain relationships.Nested structures were created, allowing for instructors and departments to be included in course documents.

4. Data Loading: The cleaned and transformed data was loaded into MongoDB collections (students, courses, and instructors) using *insert_many*.

Overall code is also present in the zip file.

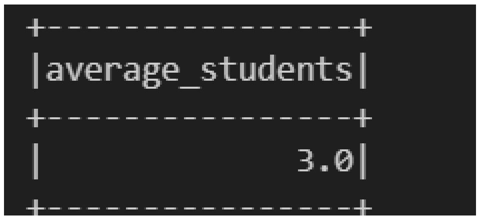
3. Query Implementation using Apache Spark

1) Fetching all students enrolled in a specific course.



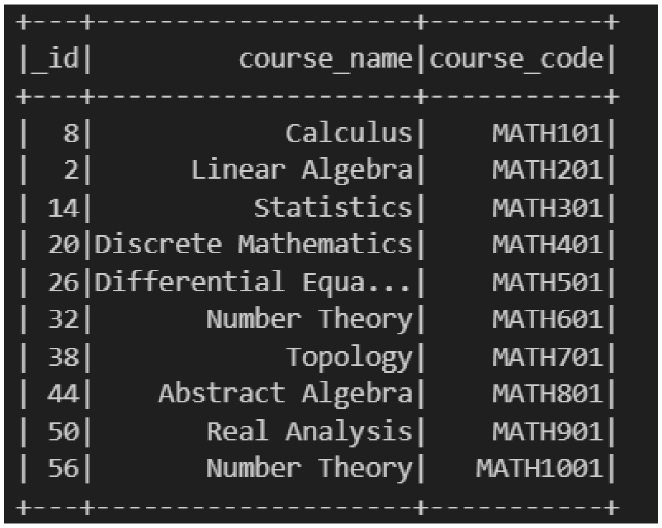
```
+---+----------+---------+
|_id|first_name|last_name|
+---+----------+---------+
| 10|       Dre|  Kersley|
|  1|    Sharyl|   Jaouen|
| 11|      Gabi|  Crystal|
| 61|    Victor|   Sleany|
| 56|      Tome|    Maven|
| 13|    Jo-ann|Heindrich|
| 57|    Nikkie|Eadmeades|
| 16|  Collette| Brenneke|
+---+----------+---------+
```

2) Calculating the average number of students enrolled in courses offered by a particular instructor at the university.



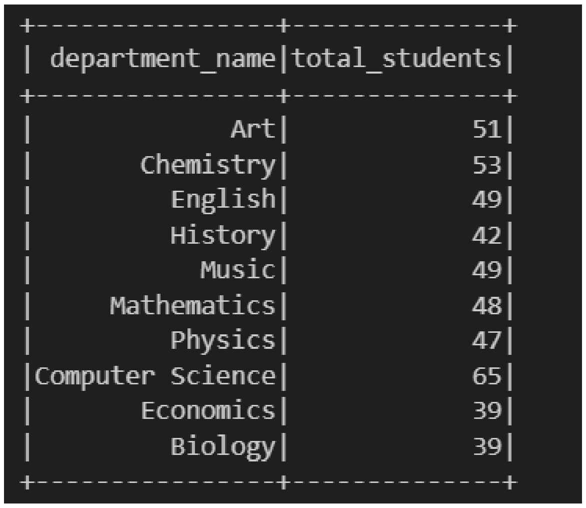
```
+----------------+
|average_students|
+----------------+
|             3.0|
+----------------+
```

3) Listing all courses offered by a specific department.



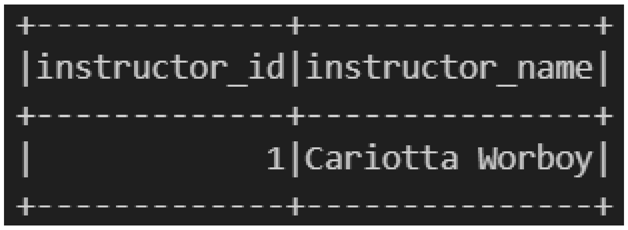
```
+---+--------------------+-----------+
|_id|         course_name|course_code|
+---+--------------------+-----------+
|  8|            Calculus|    MATH101|
|  2|      Linear Algebra|    MATH201|
| 14|          Statistics|    MATH301|
| 20|Discrete Mathematics|    MATH401|
| 26|Differential Equa...|    MATH501|
| 32|       Number Theory|    MATH601|
| 38|            Topology|    MATH701|
| 44|    Abstract Algebra|    MATH801|
| 50|       Real Analysis|    MATH901|
| 56|       Number Theory|   MATH1001|
+---+--------------------+-----------+
```

4) Finding the total number of students per department.



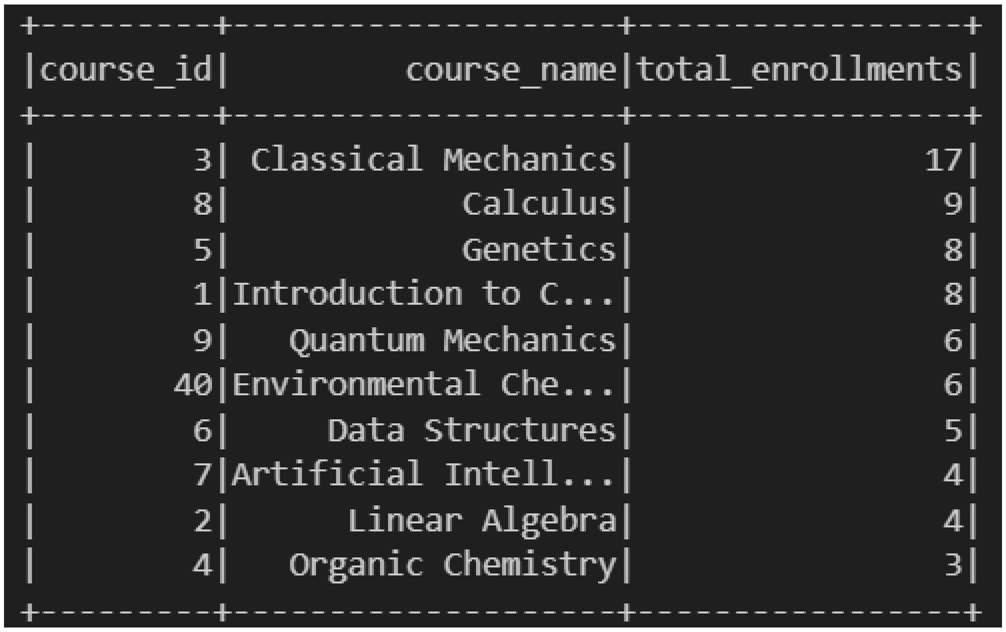
```
+----------------+--------------+
| department_name|total_students|
+----------------+--------------+
|             Art|            51|
|       Chemistry|            53|
|         English|            49|
|         History|            42|
|           Music|            49|
|     Mathematics|            48|
|         Physics|            47|
|Computer Science|            65|
|       Economics|            39|
|         Biology|            39|
+----------------+--------------+
```

5) Finding instructors who have taught all the BTech CSE core courses sometime during their tenure at the university.



```
+-------------+--------------+
|instructor_id|instructor_name|
+-------------+--------------+
|            1|Cariotta Worboy|
+-------------+--------------+
```

6) Finding top-10 courses with the highest enrollments.



```
+---------+--------------------+----------------+
|course_id|         course_name|total_enrollments|
+---------+--------------------+----------------+
|        3| Classical Mechanics|              17|
|        8|            Calculus|               9|
|        5|            Genetics|               8|
|        1|Introduction to C...|               8|
|        9|   Quantum Mechanics|               6|
|       40|Environmental Che...|               6|
|        6|     Data Structures|               5|
|        7|Artificial Intell...|               4|
|        2|      Linear Algebra|               4|
|        4|   Organic Chemistry|               3|
+---------+--------------------+----------------+
```

4. Optimizations

Techniques Used:

1. **Indexing**

```
db.instructors.createIndex({"instructor_id": 1 })
db.students.createIndex({ "student_id": 1 })
db.courses.createIndex({ "course_id": 1 })
```

db.courses.createIndex({ "course_id": 1, "instructor_id": 1 })

Indexes in MongoDB enable fast document retrieval by allowing quick access to relevant records without full collection scans. Indexing fields like `student_id`, `course_id`, and `instructor_id` decreases search times and input/output operations, enhancing performance. Compound indexes (e.g., on `course_id` and `instructor_id`) further improve efficiency for multi-field queries, reducing post-retrieval filtering. This is vital for maintaining consistent performance in read-heavy applications as the database scales

2. **Partitioning (In Spark)**

On basis of columns:

```
courses_partitioned_df = courses_df.repartition(n,"_id")
students_df = students_df.repartition(n,"_id")
instructors_partitioned_df = instructors_df.repartition(n,"_id")
```

S. No.	**Exec_time_before_optimization**	**Exec_time_after_optimization**
1	0.3738 seconds	0.2509 seconds
2	0.9351 seconds	0.4811seconds
3	0.1162 seconds	0.1156 seconds
4	0.3713 seconds	0.309 seconds
5	0.5245 seconds	0.2633seconds
6	0.3862 seconds	0.3501 seconds

It can be seen these optimizations lead to some decrease in execution time, but they can significantly perform well particularly when used with larger datasets or in distributed computing settings. However, with smaller datasets, the overhead associated with creating indexes or repartitioning may occasionally cause longer query execution times. So, overall results don't show too much of a difference!