Data Migration Process Report

1. Extract

The data extraction phase involved connecting to a PostgreSQL database and retrieving data from multiple tables:

- Departments
- Students
- Instructors
- Courses
- Enrollments

The data was extracted using SQL queries and stored in Python variables for further processing.

2. Transform

The transformation phase involved several key steps:

- Restructuring the data to fit the target MongoDB schema.
- Embedding related data (e.g., enrollments within students, courses within instructors).
- Ensuring proper data types, especially for dates.

Key transformations:

- Departments: Minimal transformation, mainly renaming fields.
- Students: Embedded enrollment data within each student record.
- Instructors: Embedded course data within each instructor record.
- Courses: Embedded enrollment data within each course record.

A custom transform_data function was implemented to handle these transformations.

3. Load

The loading phase involved inserting the transformed data into MongoDB:

1. Establishing a connection to the MongoDB database.
2. Clearing existing data (optional step).
3. Inserting transformed data into corresponding collections.

Key points:

- Used `replace_one` with `upsert=True` to handle both inserts and updates.
- Implemented error handling to manage potential insertion issues.
- Converted date objects to datetime for MongoDB compatibility.

Data Cleaning and Transformation Processes

1. Date Conversion: Implemented a `convert_dates` function to ensure all date fields are properly formatted for MongoDB.
2. Document ID Management: Used the original IDs from the source database as MongoDB document `_id`s to maintain consistency and enable easy updates.
3. Data Embedding: Restructured relational data into a document model by embedding related data (e.g., enrollments within student documents).

4. Error Handling: Implemented try-except blocks to catch and report any issues during data insertion.
5. Upsert Strategy: Used MongoDB's upsert feature to handle both new inserts and updates to existing documents, ensuring data consistency.

Results

6. The migration process successfully transferred data from PostgreSQL to MongoDB:
7. Departments: 3 documents
8. Students: 24 documents
9. Instructors: 9 documents
10. Courses: 25 documents

Queries in mongodb

- **Fetching all students enrolled in a specific course:**

  ```
  db.students.find({ "enrollments.courseId": 1 }, { firstName: 1, lastName: 1, email: 1, mobile: 1 })
  ```

  We directly filter students who have the courseId in their enrollments array, fetching essential student details

- **Calculating the average number of students enrolled in courses offered by a particular instructor**:

  ```
  db.courses.aggregate([
   { $match: { instructorId: 2 } },
   { $project: {
      courseId: 1,
      enrollmentCount: { $size: { $ifNull: ["$enrollments", []] } }
   }},
   { $group: {
      _id: "$instructorId",
      avgEnrollment: { $avg: "$enrollmentCount" }
   }}
  ])
  ```

This query matches courses by the instructorId, counts the size of the enrollments array for each course, and then calculates the average.

- Listing all courses offered by a specific department

  ```
  db.courses.find({ departmentId: 3}, { courseName: 1, instructorId: 1 })
  ```

  The query retrieves all courses where the departmentId matches, returning course names and the associated instructor.

- **Finding the total number of students per department**

  ```
  db.students.aggregate([
    { $group: {
        _id: "$departmentId",
        totalStudents: { $sum: 1 }
    }},
    { $lookup: {
        from: "departments",
        localField: "_id",
        foreignField: "departmentId",
        as: "department_info"
    }},
    { $unwind: "$department_info" },
    { $project: { department: "$department_info.departmentName", totalStudents: 1 } }
  ])
  ```

  This aggregates students by departmentId, looks up the department name from the departments collection, and returns the department name with the student count.

- **Finding instructors who have taught all the BTech CSE core courses**

  ```
        db.instructors.aggregate([
          { $match: { departmentId: 1 } },
          {
            $project: {
              firstName: 1,
              lastName: 1,
              email: 1,
  ```

```
          departmentId: 1,
          courses: {
            $filter: {
              input: "$courses",
              as: "course",
              cond: { $eq: ["$$course.departmentId", 1] }
            }
          }
        }
      }
    ])
```

- Finding top 10 courses with the highest enrollments

```
db.courses.aggregate([
  {
    $project: {
      _id: 0,
      courseId: "$_id",
      courseName: 1,
      enrollmentCount: {
        $cond: {
          if: { $isArray: "$enrollments" },
          then: { $size: "$enrollments" },
          else: 0
        }
      }
    }
  },
  {
    $sort: { enrollmentCount: -1 }
  },
  {
    $limit: 10  // Limit to top 10 courses
  }
]);
```