ASSIGNMENT - 4
CS553 - Advanced Database Systems LAB

Name: Utsav Balar
Roll no.: T24CS003

File from assignment-3: db.json

{"students": {"1": {"StudentID": 1, "Name": "John Doe", "Email": "john.doe@example.com", "Phone": "123-456-7890", "Address": "123 Main St"}, "2": {"StudentID": 2, "Name": "Jane Smith", "Email": "jane.smith@example.com", "Phone": "987-654-3210", "Address": "456 Elm St"}, "3": {"StudentID": 3, "Name": "Robert Johnson", "Email": "robert.j@example.com", "Phone": "555-123-4567", "Address": "789 Oak Ave"}, "4": {"StudentID": 4, "Name": "Emily White", "Email": "emily.white@example.com", "Phone": "111-222-3333", "Address": "567 Pine St"}, "5": {"StudentID": 5, "Name": "Michael Lee", "Email": "michael.lee@example.com", "Phone": "333-444-5555", "Address": "789 Cedar Dr"}, "6": {"StudentID": 6, "Name": "Sarah Brown", "Email": "sarah.brown@example.com", "Phone": "555-666-7777", "Address": "890 Willow Ln"}, "7": {"StudentID": 7, "Name": "David Clark", "Email": "david.clark@example.com", "Phone": "777-888-9999", "Address": "123 Birch Ave"}, "8": {"StudentID": 8, "Name": "Melissa Turner", "Email": "melissa.turner@example.com", "Phone": "888-999-0000", "Address": "456 Redwood Rd"}}, "courses": {"1": {"CourseID": 101, "CourseName": "Mathematics", "Credits": 3}, "2": {"CourseID": 102, "CourseName": "History", "Credits": 4}, "3": {"CourseID": 103, "CourseName": "Computer Science", "Credits": 3}, "4": {"CourseID": 104, "CourseName": "Literature", "Credits": 3}, "5": {"CourseID": 105, "CourseName": "Chemistry", "Credits": 4}, "6": {"CourseID": 106, "CourseName": "Physics", "Credits": 4}, "7": {"CourseID": 107, "CourseName": "Economics", "Credits": 3}, "8": {"CourseID": 108, "CourseName": "Biology", "Credits": 4}}, "exams": {"1": {"ExamID": 201, "ExamDate": "2023-11-10", "ExamTime": "09:00 AM", "Location": "Exam Hall A"}, "2": {"ExamID": 202, "ExamDate": "2023-11-12", "ExamTime": "02:00 PM", "Location": "Exam Hall B"}, "3": {"ExamID": 203, "ExamDate": "2023-11-15", "ExamTime": "10:30 AM", "Location": "Exam Hall C"}, "4": {"ExamID": 204, "ExamDate": "2023-11-18", "ExamTime": "03:15 PM", "Location": "Exam Hall D"}, "5": {"ExamID":

205, "ExamDate": "2023-11-20", "ExamTime": "01:00 PM", "Location": "Exam Hall E"}}, "faculty": {"1": {"FacultyID": 301, "Name": "Dr. Smith", "Email": "smith@example.com", "Phone": "111-222-3333", "Department": "Mathematics"}, "2": {"FacultyID": 302, "Name": "Prof. Johnson", "Email": "johnson@example.com", "Phone": "444-555-6666", "Department": "History"}, "3": {"FacultyID": 303, "Name": "Prof. Brown", "Email": "brown@example.com", "Phone": "777-888-9999", "Department": "Computer Science"}, "4": {"FacultyID": 304, "Name": "Dr. Parker", "Email": "parker@example.com", "Phone": "888-777-6666", "Department": "Chemistry"}, "5": {"FacultyID": 305, "Name": "Prof. Adams", "Email": "adams@example.com", "Phone": "999-888-7777", "Department": "Physics"}, "6": {"FacultyID": 306, "Name": "Dr. Wilson", "Email": "wilson@example.com", "Phone": "555-444-3333", "Department": "Economics"}, "7": {"FacultyID": 307, "Name": "Prof. Davis", "Email": "davis@example.com", "Phone": "333-222-1111", "Department": "Biology"}, "8": {"FacultyID": 308, "Name": "Dr. Turner", "Email": "turner@example.com", "Phone": "222-333-4444", "Department": "Literature"}}, "enrollment": {"1": {"EnrollmentID": 1, "StudentID": 1, "CourseID": 101, "EnrollmentDate": "2023-09-01"}, "2": {"EnrollmentID": 2, "StudentID": 1, "CourseID": 102, "EnrollmentDate": "2023-09-01"}, "3": {"EnrollmentID": 3, "StudentID": 2, "CourseID": 101, "EnrollmentDate": "2023-09-02"}, "4": {"EnrollmentID": 4, "StudentID": 3, "CourseID": 103, "EnrollmentDate": "2023-09-03"}, "5": {"EnrollmentID": 5, "StudentID": 4, "CourseID": 104, "EnrollmentDate": "2023-09-04"}, "6": {"EnrollmentID": 6, "StudentID": 5, "CourseID": 105, "EnrollmentDate": "2023-09-05"}, "7": {"EnrollmentID": 7, "StudentID": 6, "CourseID": 106, "EnrollmentDate": "2023-09-06"}, "8": {"EnrollmentID": 8, "StudentID": 7, "CourseID": 107, "EnrollmentDate": "2023-09-07"}, "9": {"EnrollmentID": 9, "StudentID": 8, "CourseID": 108, "EnrollmentDate": "2023-09-08"}}, "teaching": {"1": {"TeachingID": 1, "FacultyID": 301, "CourseID": 101}, "2": {"TeachingID": 2, "FacultyID": 302, "CourseID": 102}, "3": {"TeachingID": 3, "FacultyID": 303, "CourseID": 103}, "4": {"TeachingID": 4, "FacultyID": 304, "CourseID": 104}, "5": {"TeachingID": 5, "FacultyID": 305, "CourseID": 105}, "6": {"TeachingID": 6, "FacultyID": 306, "CourseID": 106}, "7": {"TeachingID": 7, "FacultyID": 307, "CourseID": 107}, "8": {"TeachingID": 8, "FacultyID": 308, "CourseID": 108}}, "exam_registration": {"1":

{"RegistrationID": 101, "StudentID": 1, "ExamID": 201, "RegistrationDate": "2023-10-15"}, "2": {"RegistrationID": 102, "StudentID": 2, "ExamID": 201, "RegistrationDate": "2023-10-16"}, "3": {"RegistrationID": 103, "StudentID": 3, "ExamID": 202, "RegistrationDate": "2023-10-17"}, "4": {"RegistrationID": 104, "StudentID": 4, "ExamID": 203, "RegistrationDate": "2023-10-18"}, "5": {"RegistrationID": 105, "StudentID": 5, "ExamID": 204, "RegistrationDate": "2023-10-19"}, "6": {"RegistrationID": 106, "StudentID": 6, "ExamID": 205, "RegistrationDate": "2023-10-20"}, "7": {"RegistrationID": 107, "StudentID": 7, "ExamID": 201, "RegistrationDate": "2023-10-21"}, "8": {"RegistrationID": 108, "StudentID": 8, "ExamID": 202, "RegistrationDate": "2023-10-22"}}, "exam_results": {"1": {"ResultID": 501, "StudentID": 1, "ExamID": 201, "Score": 92.5}, "2": {"ResultID": 502, "StudentID": 2, "ExamID": 201, "Score": 88.0}, "3": {"ResultID": 503, "StudentID": 3, "ExamID": 202, "Score": 95.5}, "4": {"ResultID": 504, "StudentID": 4, "ExamID": 203, "Score": 89.0}, "5": {"ResultID": 505, "StudentID": 5, "ExamID": 204, "Score": 94.5}, "6": {"ResultID": 506, "StudentID": 6, "ExamID": 205, "Score": 91.0}, "7": {"ResultID": 507, "StudentID": 7, "ExamID": 201, "Score": 87.5}}}

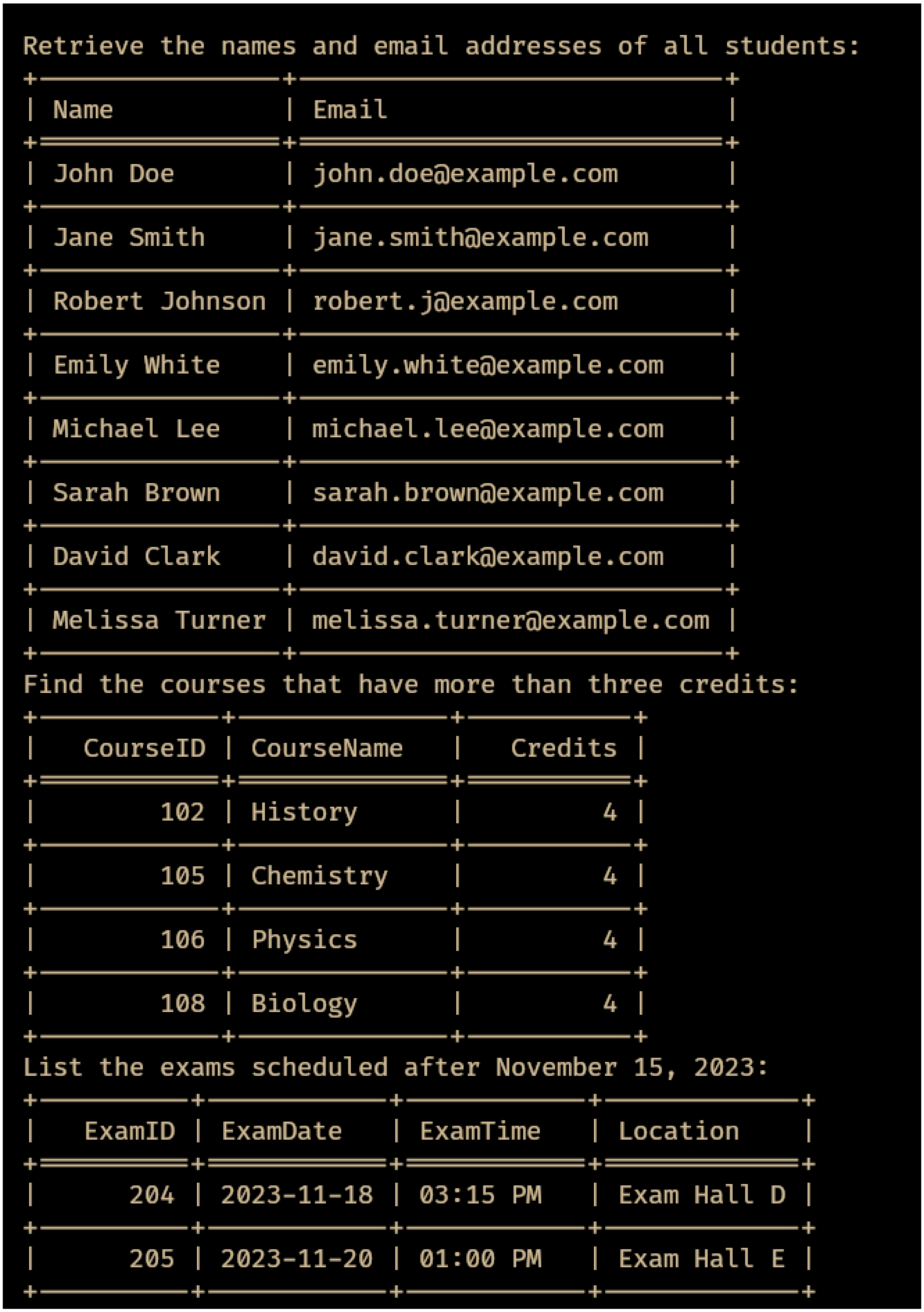
File: t24cs003-assignment-4.py

```
from tinydb import TinyDB, Query
from statistics import mean
from tabulate import tabulate

db = TinyDB("db.json")

students = db.table("students")
all_students = students.all()
student_names_emails = [(s["Name"], s["Email"]) for s in all_students]
print("Retrieve the names and email addresses of all students:")
print(tabulate(student_names_emails, headers=["Name", "Email"],
tablefmt="grid"))

courses = db.table("courses")
high_credit_courses = courses.search(Query().Credits > 3)
print("Find the courses that have more than three credits:")
print(
    tabulate(
             [(c["CourseID"], c["CourseName"], c["Credits"]) for c in
high_credit_courses],
        headers=["CourseID", "CourseName", "Credits"],
        tablefmt="grid",
    )
)

exams = db.table("exams")
scheduled_exams = exams.search(Query().ExamDate > "2023-11-15")
print("List the exams scheduled after November 15, 2023:")
print(
    tabulate(
        [
            (e["ExamID"], e["ExamDate"], e["ExamTime"], e["Location"])
            for e in scheduled_exams
        ],
        headers=["ExamID", "ExamDate", "ExamTime", "Location"],
        tablefmt="grid",
    )
)

faculty = db.table("faculty")
math_faculty = faculty.search(Query().Department == "Mathematics")
print("Get the faculty members who work in the 'Mathematics'
department:")
print(
    tabulate(
        [
                    (f["FacultyID"], f["Name"], f["Email"], f["Phone"],
f["Department"])
            for f in math_faculty
        ],
        headers=["FacultyID", "Name", "Email", "Phone", "Department"],
        tablefmt="grid",
    )
)

enrollment = db.table("enrollment")
student_courses = [(e["StudentID"], e["CourseID"]) for e in
enrollment.all()]
print("Retrieve the courses that each student is enrolled in:")
print(tabulate(student_courses,    headers=["StudentID",    "CourseID"],
tablefmt="grid"))

exam_results = db.table("exam_results")
exam_ids = {r["ExamID"] for r in exam_results.all()}
average_scores = {
    exam_id: mean([r["Score"] for r in exam_results.search(Query().ExamID
== exam_id)])
    for exam_id in exam_ids
}
print("Find the average score for each exam:")
print(
    tabulate(
        [(exam_id, avg) for exam_id, avg in average_scores.items()],
        headers=["ExamID", "Average Score"],
        tablefmt="grid",
    )
)

high_scorers = {r["StudentID"] for r in exam_results.search(Query().Score
> 90)}
print("List the students who scored above 90 on any exam:")
print(tabulate([(s,)  for  s  in  high_scorers],  headers=["StudentID"],
tablefmt="grid"))

teaching = db.table("teaching")
faculty_teach_counts = {}
for t in teaching.all():
    faculty_teach_counts[t["FacultyID"]] = (
        faculty_teach_counts.get(t["FacultyID"], 0) + 1
    )
multi_course_faculty = [f for f, count in faculty_teach_counts.items() if
count > 1]
print("Retrieve the faculty members who teach multiple courses:")
print(
    tabulate(
           [(f,)  for  f  in  multi_course_faculty],  headers=["FacultyID"],
tablefmt="grid"
    )
)

exam_registration = db.table("exam_registration")
registered_students = {r["StudentID"] for r in exam_registration.all()}
```

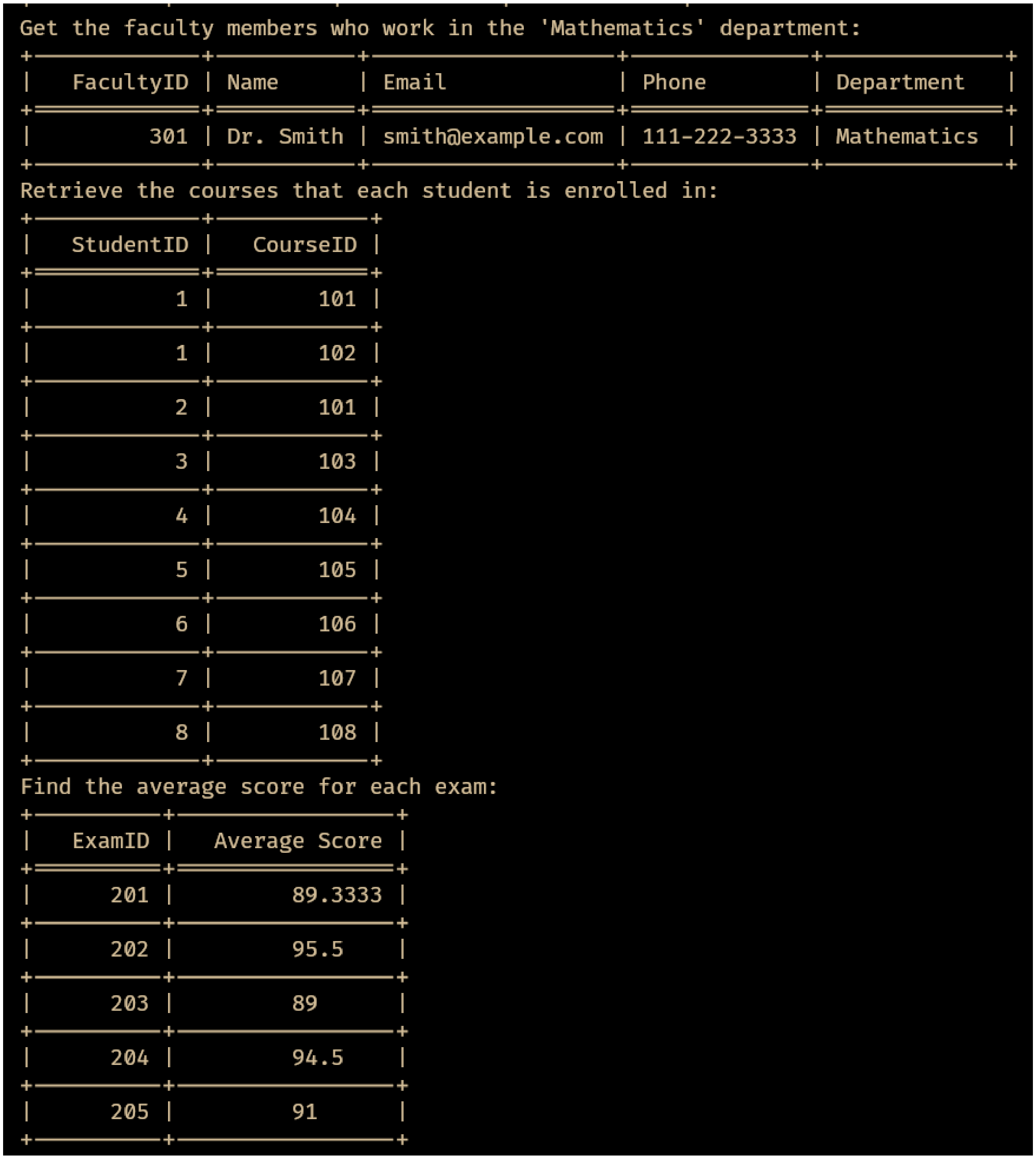
```
unregistered_students = [
      s["StudentID"] for s in all_students if s["StudentID"] not in
registered_students
]
print("Find the students who have not registered for any exams:")
print(
    tabulate(
          [(s,) for s in unregistered_students], headers=["StudentID"],
tablefmt="grid"
    )
)

course_enrollment_counts = {}
for e in enrollment.all():
    course_enrollment_counts[e["CourseID"]] = (
        course_enrollment_counts.get(e["CourseID"], 0) + 1
    )
print("Retrieve the total number of enrollments for each course:")
print(
    tabulate(
                      [(course_id, count) for course_id, count in
course_enrollment_counts.items()],
        headers=["CourseID", "Enrollments"],
        tablefmt="grid",
    )
)

history_course = courses.get(Query().CourseName == "History")
history_students = [
    e["StudentID"]
                for e in enrollment.search(Query().CourseID ==
history_course["CourseID"])
]
print("Find the students who are enrolled in the 'History' course:")
print(
      tabulate([(s,) for s in history_students], headers=["StudentID"],
tablefmt="grid")
)

exams = db.table("exams")
november_exams = exams.search(
     (Query().ExamDate >= "2023-11-01") & (Query().ExamDate <= "2023-11-
30")
```

```
)
november_exams_locations = [(e["ExamDate"], e["Location"]) for e in
november_exams]
print("Retrieve the exams and their locations scheduled for November
2023:")
print(
    tabulate(
             november_exams_locations, headers=["ExamDate", "Location"],
tablefmt="grid"
    )
)

course_enrollment_counts = {}
for e in enrollment.all():
    course_enrollment_counts[e["CourseID"]] = (
        course_enrollment_counts.get(e["CourseID"], 0) + 1
    )
most_enrolled_course = max(course_enrollment_counts,
key=course_enrollment_counts.get)
print("List the courses with the highest number of enrollments:")
print(
    tabulate(
                                                    [(most_enrolled_course,
course_enrollment_counts[most_enrolled_course])],
        headers=["CourseID", "Enrollments"],
        tablefmt="grid",
    )
)

exam_results = db.table("exam_results")
student_scores = {}
for r in exam_results.all():
      student_scores[r["StudentID"]] = student_scores.get(r["StudentID"],
[]) + [
        r["Score"]
    ]
average_student_scores = {s: mean(scores) for s, scores in
student_scores.items()}
print("Find the average score for each student:")
print(
    tabulate(
        [(s, avg) for s, avg in average_student_scores.items()],
        headers=["StudentID", "Average Score"],
        tablefmt="grid",
    )
)

exam_registration = db.table("exam_registration")
exam_ids_with_registrations = {r["ExamID"] for r in
exam_registration.all()}
unregistered_exams = [
         e["ExamID"] for e in exams.all() if e["ExamID"] not in
exam_ids_with_registrations
]
print("Retrieve the exams that have no registered students:")
print(tabulate([(e,) for e in unregistered_exams], headers=["ExamID"],
tablefmt="grid"))

faculty = db.table("faculty")
teaching = db.table("teaching")

taught_faculty_ids = {t["FacultyID"] for t in teaching.all()}
untaught_faculty = [
    f for f in faculty.all() if f["FacultyID"] not in taught_faculty_ids
]
print("List the faculty members who have yet to teach any courses:")
print(
    tabulate(
        [(f["FacultyID"], f["Name"]) for f in untaught_faculty],
        headers=["FacultyID", "Name"],
        tablefmt="grid",
    )
)

math_course = courses.get(Query().CourseName == "Mathematics")
["CourseID"]
cs_course = courses.get(Query().CourseName == "Computer Science")
["CourseID"]
math_students = {
        e["StudentID"] for e in enrollment.search(Query().CourseID ==
math_course)
}
cs_students = {e["StudentID"] for e in enrollment.search(Query().CourseID
== cs_course)}
students_in_both = math_students.intersection(cs_students)
print(
        "Find the students who have registered for exams in both
'Mathematics' and 'Computer Science' departments:"
)
print(
      tabulate([(s,) for s in students_in_both], headers=["StudentID"],
tablefmt="grid")
)

highest_scores = {}
for r in exam_results.all():
    exam_id = r["ExamID"]
            if exam_id not in highest_scores or r["Score"] >
highest_scores[exam_id]["Score"]:
        highest_scores[exam_id] = r
highest_student_scores = {s["StudentID"]: s["Score"] for s in
highest_scores.values()}
print("Retrieve the students who scored the highest in each exam:")
print(
    tabulate(
        [(s, score) for s, score in highest_student_scores.items()],
        headers=["StudentID", "Highest Score"],
        tablefmt="grid",
    )
)

enrolled_course_ids = {e["CourseID"] for e in enrollment.all()}
unenrolled_courses = [
       c["CourseID"] for c in courses.all() if c["CourseID"] not in
enrolled_course_ids
]
print("Find the courses that no student has enrolled in:")
print(
     tabulate([(c,) for c in unenrolled_courses], headers=["CourseID"],
tablefmt="grid")
)

high_enrollment_courses = [
    course_id for course_id, count in course_enrollment_counts.items() if
count > 10
]
high_enrollment_faculty = [
    t["FacultyID"]
```

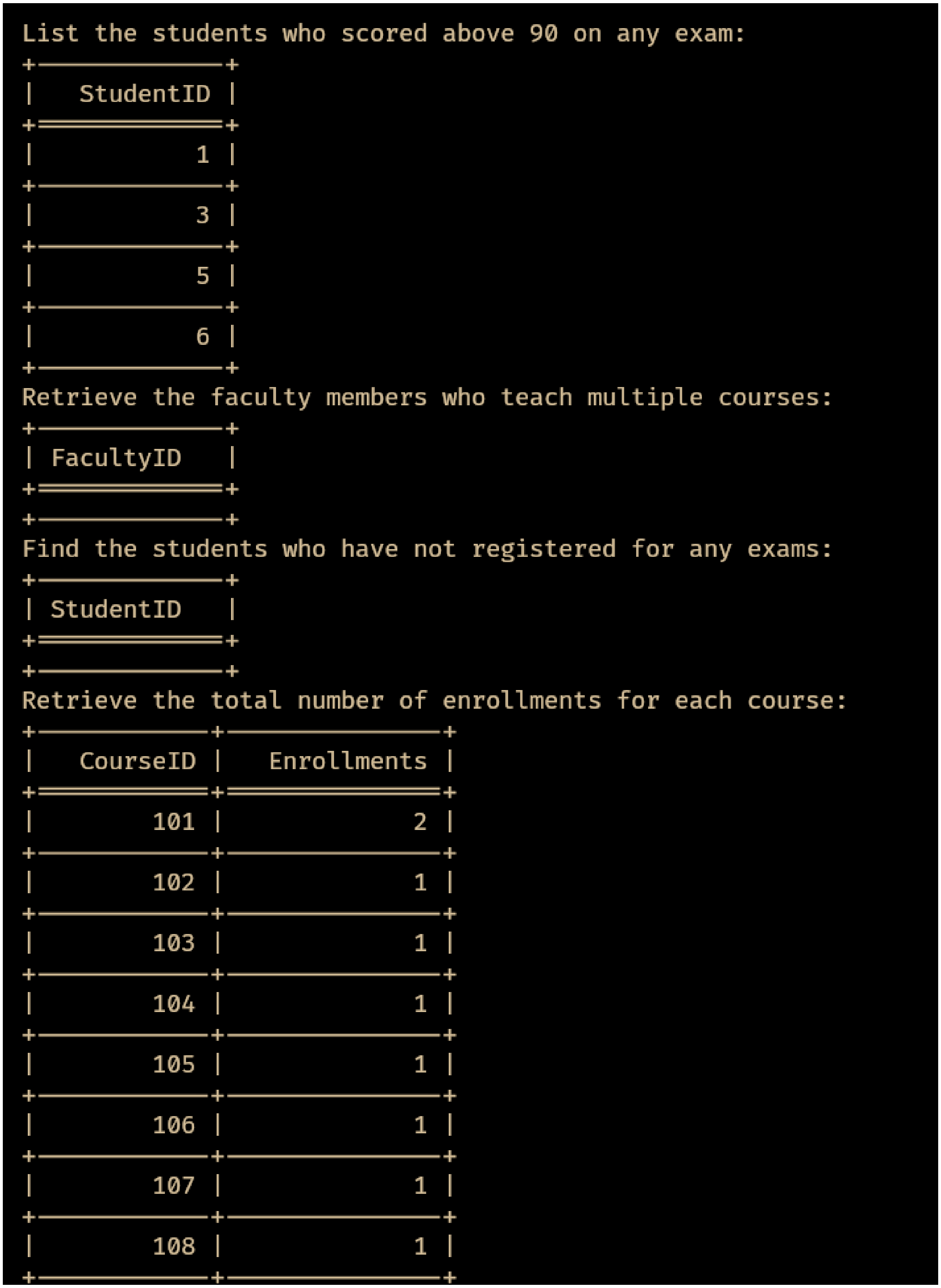
```
                                                    for            t            in
teaching.search(Query().CourseID.one_of(high_enrollment_courses))
]
print(
      "Retrieve the faculty members who teach courses with an average
enrollment count above 10:"
)
print(
    tabulate(
          [(f,) for f in high_enrollment_faculty], headers=["FacultyID"],
tablefmt="grid"
    )
)
```



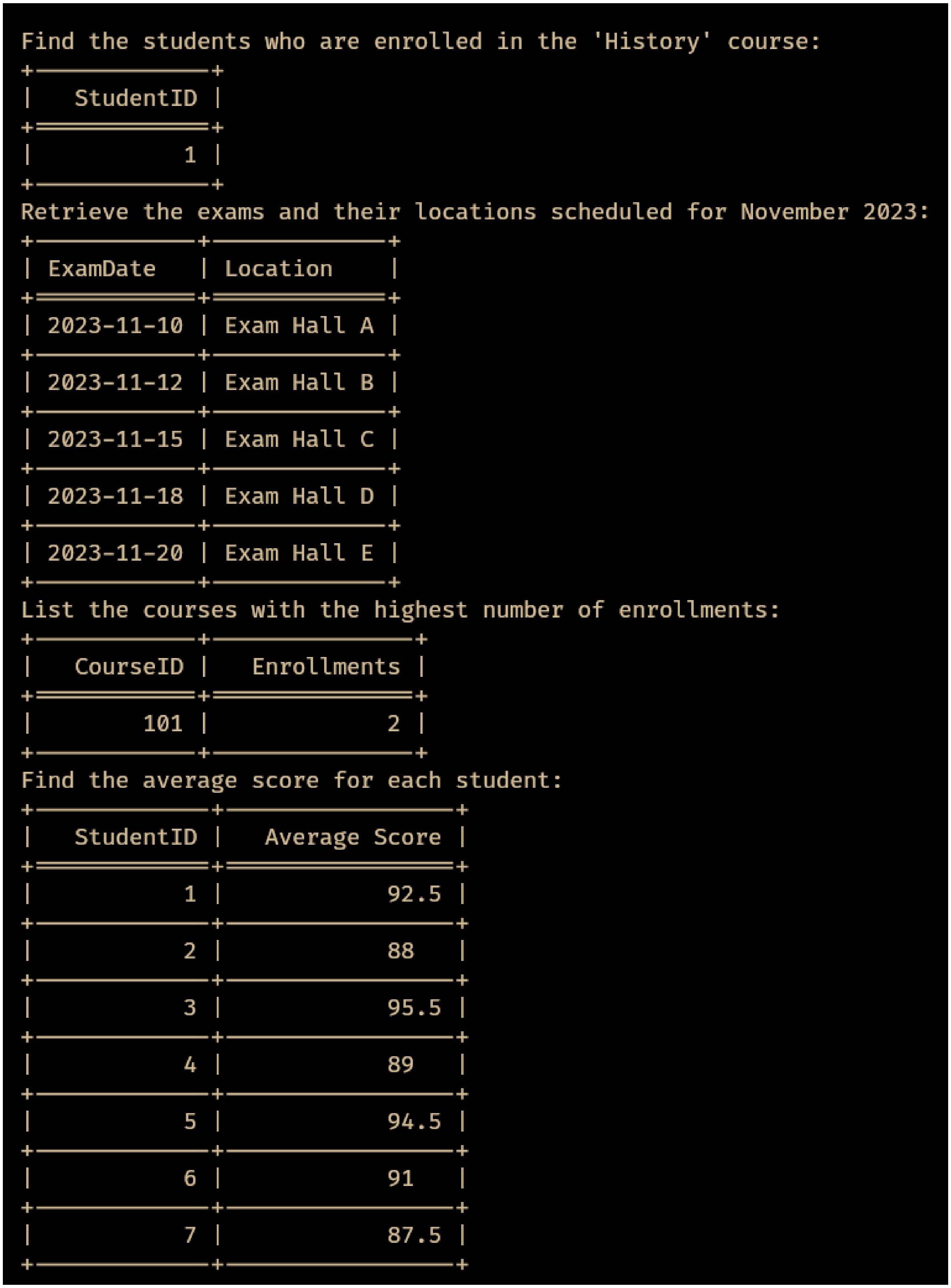
```
Retrieve the names and email addresses of all students:
+----------------+----------------------------+
| Name           | Email                      |
+================+============================+
| John Doe       | john.doe@example.com       |
+----------------+----------------------------+
| Jane Smith     | jane.smith@example.com     |
+----------------+----------------------------+
| Robert Johnson | robert.j@example.com       |
+----------------+----------------------------+
| Emily White    | emily.white@example.com    |
+----------------+----------------------------+
| Michael Lee    | michael.lee@example.com    |
+----------------+----------------------------+
| Sarah Brown    | sarah.brown@example.com    |
+----------------+----------------------------+
| David Clark    | david.clark@example.com    |
+----------------+----------------------------+
| Melissa Turner | melissa.turner@example.com |
+----------------+----------------------------+
Find the courses that have more than three credits:
+------------+--------------+-----------+
|   CourseID | CourseName   |   Credits |
+============+==============+===========+
|        102 | History      |         4 |
+------------+--------------+-----------+
|        105 | Chemistry    |         4 |
+------------+--------------+-----------+
|        106 | Physics      |         4 |
+------------+--------------+-----------+
|        108 | Biology      |         4 |
+------------+--------------+-----------+
List the exams scheduled after November 15, 2023:
+----------+------------+------------+-------------+
|   ExamID | ExamDate   | ExamTime   | Location    |
+==========+============+============+=============+
|      204 | 2023-11-18 | 03:15 PM   | Exam Hall D |
+----------+------------+------------+-------------+
|      205 | 2023-11-20 | 01:00 PM   | Exam Hall E |
+----------+------------+------------+-------------+
```



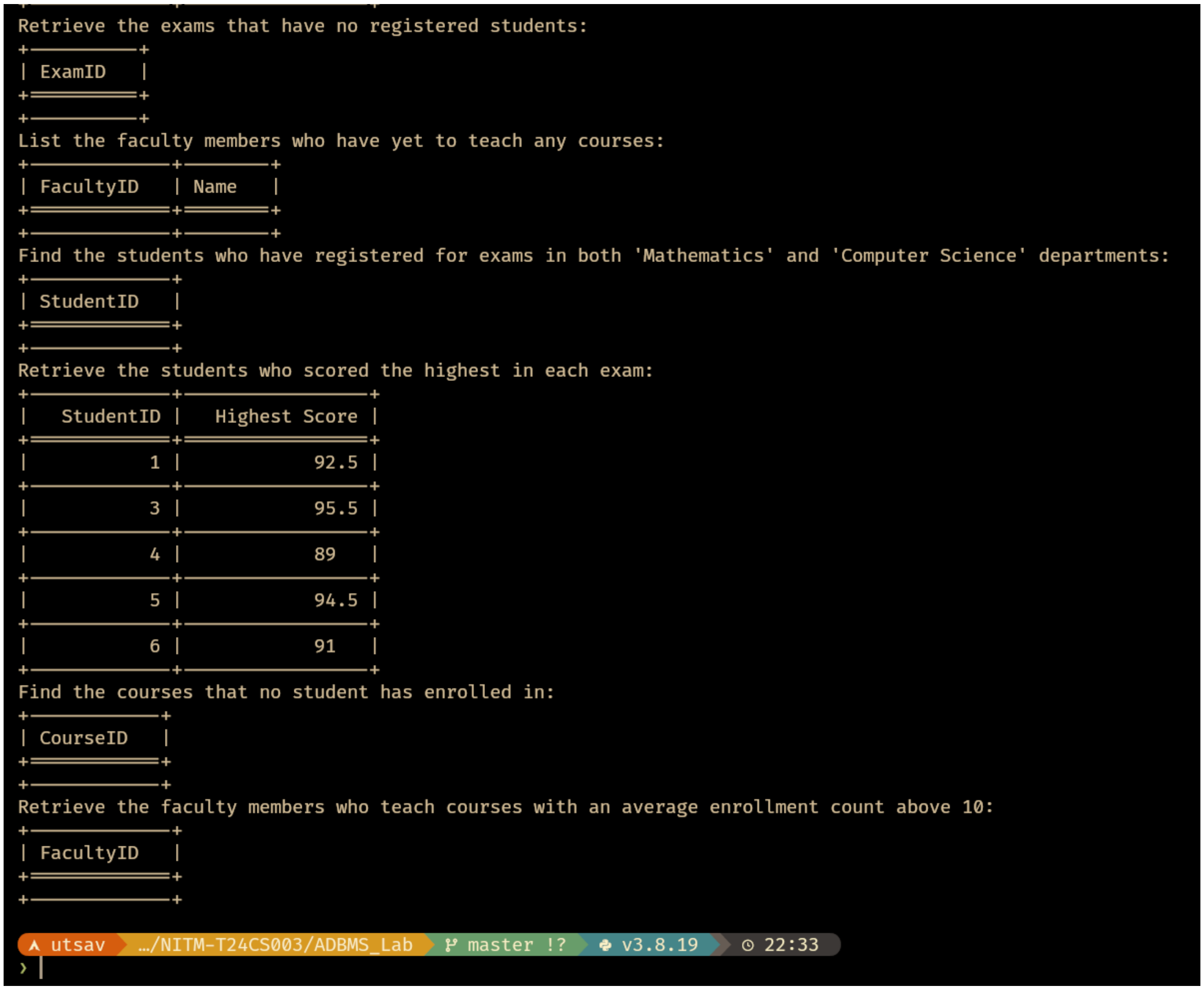
```
Get the faculty members who work in the 'Mathematics' department:
+-------------+------------+-------------------+--------------+--------------+
|   FacultyID | Name       | Email             | Phone        | Department   |
+=============+============+===================+==============+==============+
|         301 | Dr. Smith  | smith@example.com | 111-222-3333 | Mathematics  |
+-------------+------------+-------------------+--------------+--------------+
Retrieve the courses that each student is enrolled in:
+-------------+------------+
|   StudentID |   CourseID |
+=============+============+
|           1 |        101 |
+-------------+------------+
|           1 |        102 |
+-------------+------------+
|           2 |        101 |
+-------------+------------+
|           3 |        103 |
+-------------+------------+
|           4 |        104 |
+-------------+------------+
|           5 |        105 |
+-------------+------------+
|           6 |        106 |
+-------------+------------+
|           7 |        107 |
+-------------+------------+
|           8 |        108 |
+-------------+------------+
Find the average score for each exam:
+----------+-----------------+
|   ExamID |   Average Score |
+==========+=================+
|      201 |         89.3333 |
+----------+-----------------+
|      202 |         95.5    |
+----------+-----------------+
|      203 |         89      |
+----------+-----------------+
|      204 |         94.5    |
+----------+-----------------+
|      205 |         91      |
+----------+-----------------+
```



```
List the students who scored above 90 on any exam:
+─────────────+
|   StudentID |
+═════════════+
|           1 |
+─────────────+
|           3 |
+─────────────+
|           5 |
+─────────────+
|           6 |
+─────────────+
Retrieve the faculty members who teach multiple courses:
+─────────────+
| FacultyID   |
+═════════════+
+─────────────+
Find the students who have not registered for any exams:
+─────────────+
| StudentID   |
+═════════════+
+─────────────+
Retrieve the total number of enrollments for each course:
+────────────+───────────────+
|   CourseID |   Enrollments |
+════════════+═══════════════+
|        101 |             2 |
+────────────+───────────────+
|        102 |             1 |
+────────────+───────────────+
|        103 |             1 |
+────────────+───────────────+
|        104 |             1 |
+────────────+───────────────+
|        105 |             1 |
+────────────+───────────────+
|        106 |             1 |
+────────────+───────────────+
|        107 |             1 |
+────────────+───────────────+
|        108 |             1 |
+────────────+───────────────+
```



```
Find the students who are enrolled in the 'History' course:
+-------------+
|   StudentID |
+=============+
|           1 |
+-------------+
Retrieve the exams and their locations scheduled for November 2023:
+------------+-------------+
| ExamDate   | Location    |
+============+=============+
| 2023-11-10 | Exam Hall A |
+------------+-------------+
| 2023-11-12 | Exam Hall B |
+------------+-------------+
| 2023-11-15 | Exam Hall C |
+------------+-------------+
| 2023-11-18 | Exam Hall D |
+------------+-------------+
| 2023-11-20 | Exam Hall E |
+------------+-------------+
List the courses with the highest number of enrollments:
+------------+---------------+
|   CourseID |   Enrollments |
+============+===============+
|        101 |             2 |
+------------+---------------+
Find the average score for each student:
+-------------+-----------------+
|   StudentID |   Average Score |
+=============+=================+
|           1 |            92.5 |
+-------------+-----------------+
|           2 |            88   |
+-------------+-----------------+
|           3 |            95.5 |
+-------------+-----------------+
|           4 |            89   |
+-------------+-----------------+
|           5 |            94.5 |
+-------------+-----------------+
|           6 |            91   |
+-------------+-----------------+
|           7 |            87.5 |
+-------------+-----------------+
```



```
Retrieve the exams that have no registered students:
+----------+
| ExamID   |
+==========+
+----------+
List the faculty members who have yet to teach any courses:
+-------------+---------+
| FacultyID   | Name    |
+=============+=========+
+-------------+---------+
Find the students who have registered for exams in both 'Mathematics' and 'Computer Science' departments:
+-------------+
| StudentID   |
+=============+
+-------------+
Retrieve the students who scored the highest in each exam:
+-------------+-----------------+
|   StudentID |   Highest Score |
+=============+=================+
|           1 |            92.5 |
+-------------+-----------------+
|           3 |            95.5 |
+-------------+-----------------+
|           4 |            89   |
+-------------+-----------------+
|           5 |            94.5 |
+-------------+-----------------+
|           6 |            91   |
+-------------+-----------------+
Find the courses that no student has enrolled in:
+------------+
| CourseID   |
+============+
+------------+
Retrieve the faculty members who teach courses with an average enrollment count above 10:
+-------------+
| FacultyID   |
+=============+
+-------------+

 utsav  …/NITM-T24CS003/ADBMS_Lab  master !?  v3.8.19  22:33
›
```