

1. CourseInfo

목적: 특정 강의에 대한 기본 정보를 조회합니다.

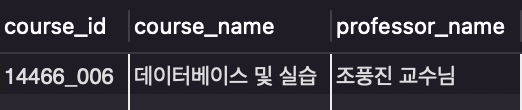
구성:

course\_id: 강의 ID

course\_name: 강의 이름

professor\_name: 교수 이름

특징: course\_id가 '14466\_006'인 강의에 대한 정보를 제공합니다.



2. CourseEvaluationInfo

목적: 특정 강의에 대한 평가 정보를 집계하여 제공하는 뷰입니다.

구성:

course\_id: 강의 ID

average\_rating: 평균 평점 (소수점 2자리)

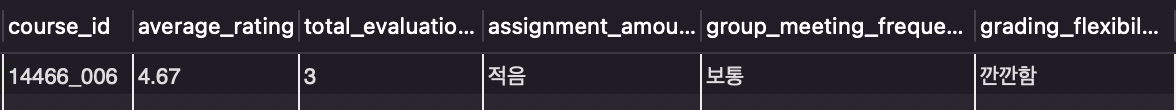
total\_evaluations: 총 평가 수

assignment\_amount: 과제 양에 대한 평가 (많음, 보통, 적음)

group\_meeting\_frequency: 조모임 빈도에 대한 평가 (많음, 보통, 적음)

grading\_flexibility: 성적 기준에 대한 평가 (너그러움, 보통, 깐깐함)

특징: course\_id가 '14466\_006'인 강의에 대한 다양한 평가 지표를 제공합니다.



3. CourseEvaluationDetails

목적: 개별 강의평의 세부 정보를 집계하여 제공합니다.

구성:

course\_evaluation\_id: 강의평 ID

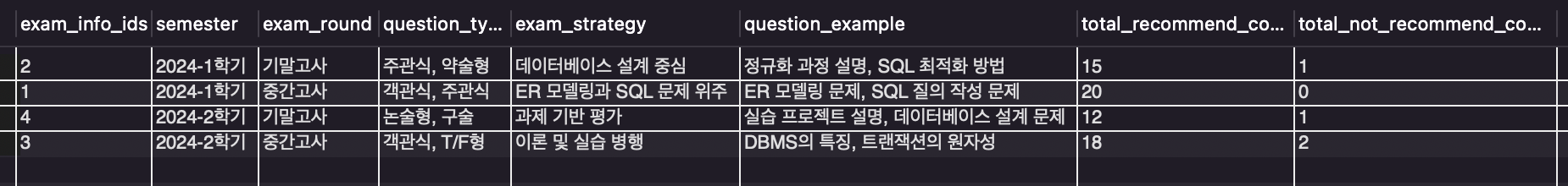
average\_rating: 평균 평점

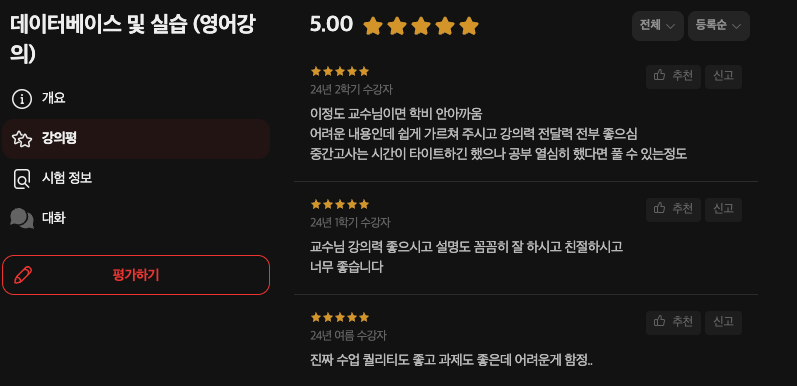
semesters\_attended: 수강한 학기 수

course\_evaluation: 강의평 내용 (여러 내용을 '; '로 구분)

total\_recommend\_count: 총 추천 수

특징: 각 강의평에 대한 집계 정보를 제공하며, 강의평 ID로 그룹화합니다.





4. ExamInfoDetails

목적: 시험 정보에 대한 세부 사항을 집계하여 제공합니다.

구성:

exam\_info\_ids: 시험 정보 ID (중복 제거)

semester: 학기 정보

exam\_round: 시험 회차 (중간고사, 기말고사)

question\_type: 문제 유형 (중복 제거)

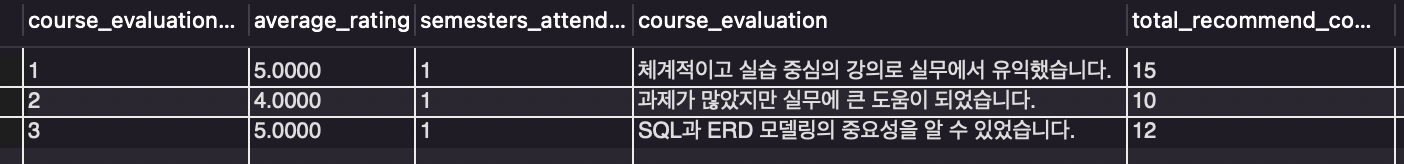
exam\_strategy: 시험 전략 (중복 제거)

question\_example: 문제 예시 (중복 제거)

total\_recommend\_count: 총 추천 수

total\_not\_recommend\_count: 총 비추천 수

특징: 학기와 시험 회차로 그룹화하여 시험 관련 정보를 제공합니다.



**[선택]**

5. UserSanctionDetails

목적: 사용자 제재에 대한 정보를 제공합니다.

구성:

sanction\_id: 제재 ID

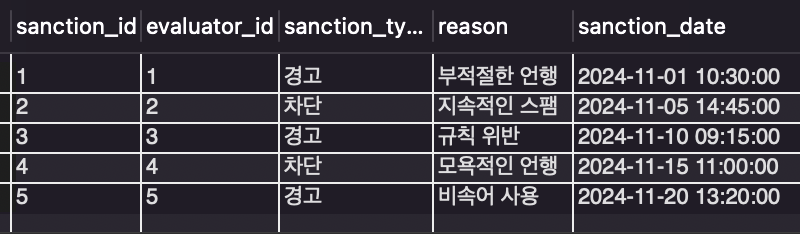
evaluator\_id: 신고를 당한 유저 ID

sanction\_type: 제재 종류

reason: 제재 사유

sanction\_date: 제재 일시

특징: 제재와 관련된 정보를 상세히 제공합니다.



PYTHON

from flask import Flask, request, jsonify

import pymysql

import pandas as pd

app = Flask(\_\_name\_\_)

# *데이터베이스 연결 설정*

def get\_db\_connection():

return pymysql.connect(

host='localhost',

port=3306,

user='root',

password='q9412201',

db='everytime'

)

@app.route('/course\_info', methods=['POST'])

def get\_course\_info():

request\_json = request.get\_json()

course\_id = request\_json['course\_id']

sql\_query = f"""

SELECT \* FROM CourseInfo WHERE course\_id = '{course\_id}';

"""

try:

conn = get\_db\_connection()

df\_course\_info = pd.read\_sql\_query(sql\_query, conn)

except Exception as e:

return jsonify({"error": str(e)})

finally:

conn.close()

return jsonify(df\_course\_info.to\_dict(orient='records'))

@app.route('/course\_evaluation\_info', methods=['POST'])

def get\_course\_evaluation\_info():

request\_json = request.get\_json()

course\_id = request\_json['course\_id']

sql\_query = f"""

SELECT \* FROM CourseEvaluationInfo WHERE course\_id = '{course\_id}';

"""

try:

conn = get\_db\_connection()

df\_evaluation\_info = pd.read\_sql\_query(sql\_query, conn)

except Exception as e:

return jsonify({"error": str(e)})

finally:

conn.close()

return jsonify(df\_evaluation\_info.to\_dict(orient='records'))

@app.route('/course\_evaluation\_details', methods=['GET'])

def get\_course\_evaluation\_details():

sql\_query = "SELECT \* FROM CourseEvaluationDetails;"

try:

conn = get\_db\_connection()

df\_evaluation\_details = pd.read\_sql\_query(sql\_query, conn)

except Exception as e:

return jsonify({"error": str(e)})

finally:

conn.close()

return jsonify(df\_evaluation\_details.to\_dict(orient='records'))

@app.route('/exam\_info\_details', methods=['GET'])

def get\_exam\_info\_details():

sql\_query = "SELECT \* FROM ExamInfoDetails;"

try:

conn = get\_db\_connection()

df\_exam\_info\_details = pd.read\_sql\_query(sql\_query, conn)

except Exception as e:

return jsonify({"error": str(e)})

finally:

conn.close()

return jsonify(df\_exam\_info\_details.to\_dict(orient='records'))

@app.route('/user\_sanction\_details', methods=['GET'])

def get\_user\_sanction\_details():

sql\_query = "SELECT \* FROM UserSanctionDetails;"

try:

conn = get\_db\_connection()

df\_user\_sanction\_details = pd.read\_sql\_query(sql\_query, conn)

except Exception as e:

return jsonify({"error": str(e)})

finally:

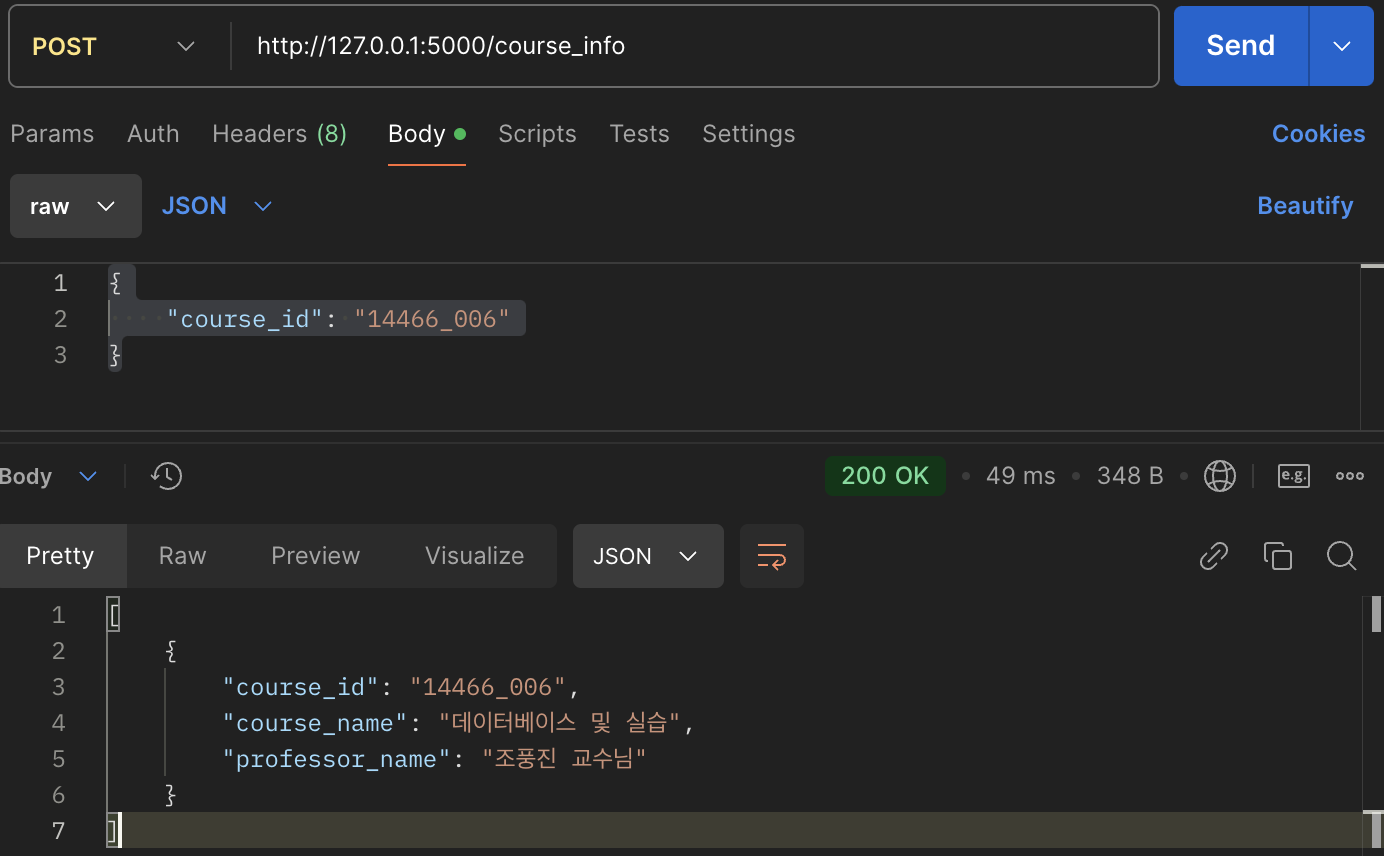
conn.close()

return jsonify(df\_user\_sanction\_details.to\_dict(orient='records'))

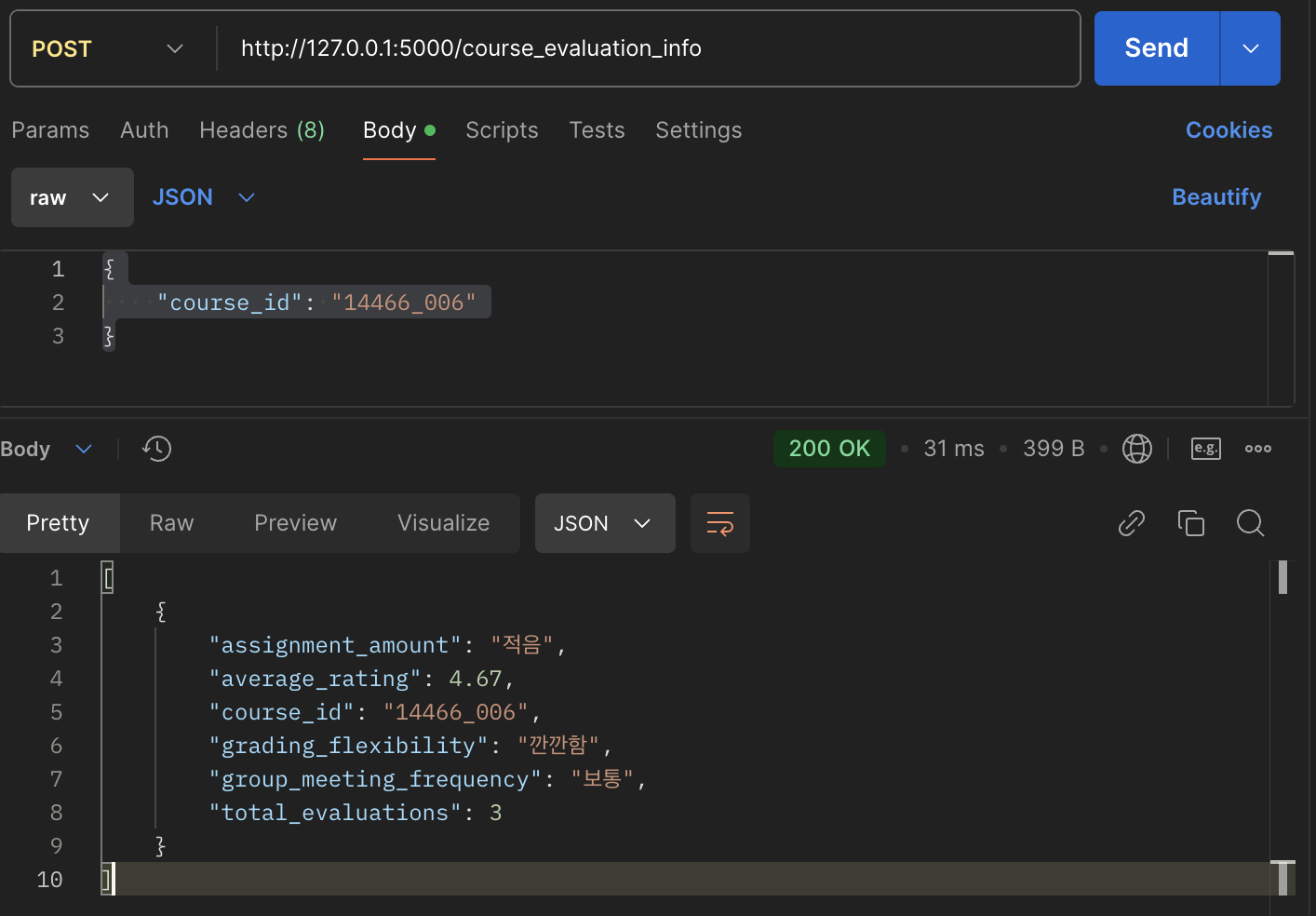
if \_\_name\_\_ == "\_\_main\_\_":

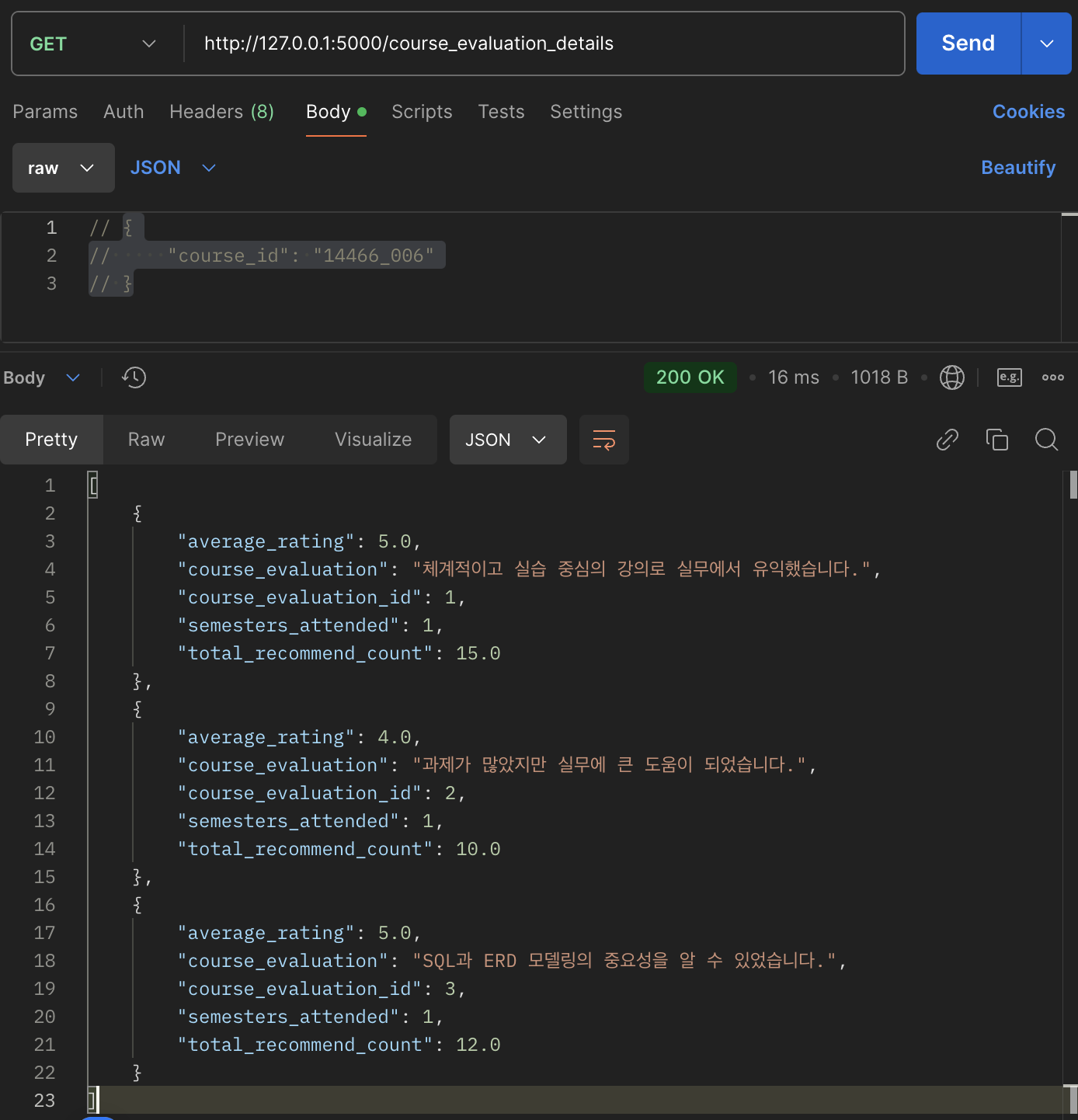
app.run(debug=True)

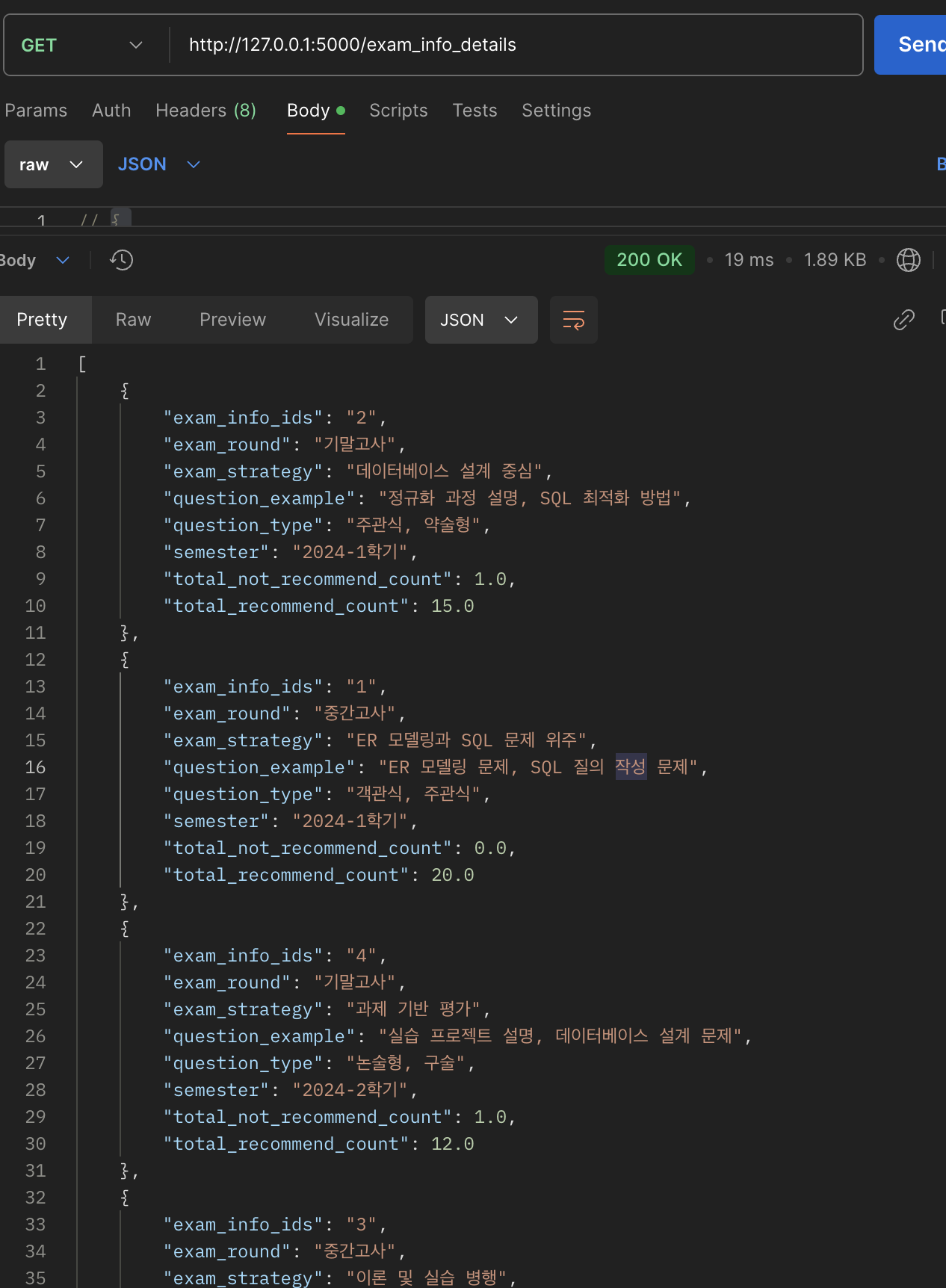
POSTMAN

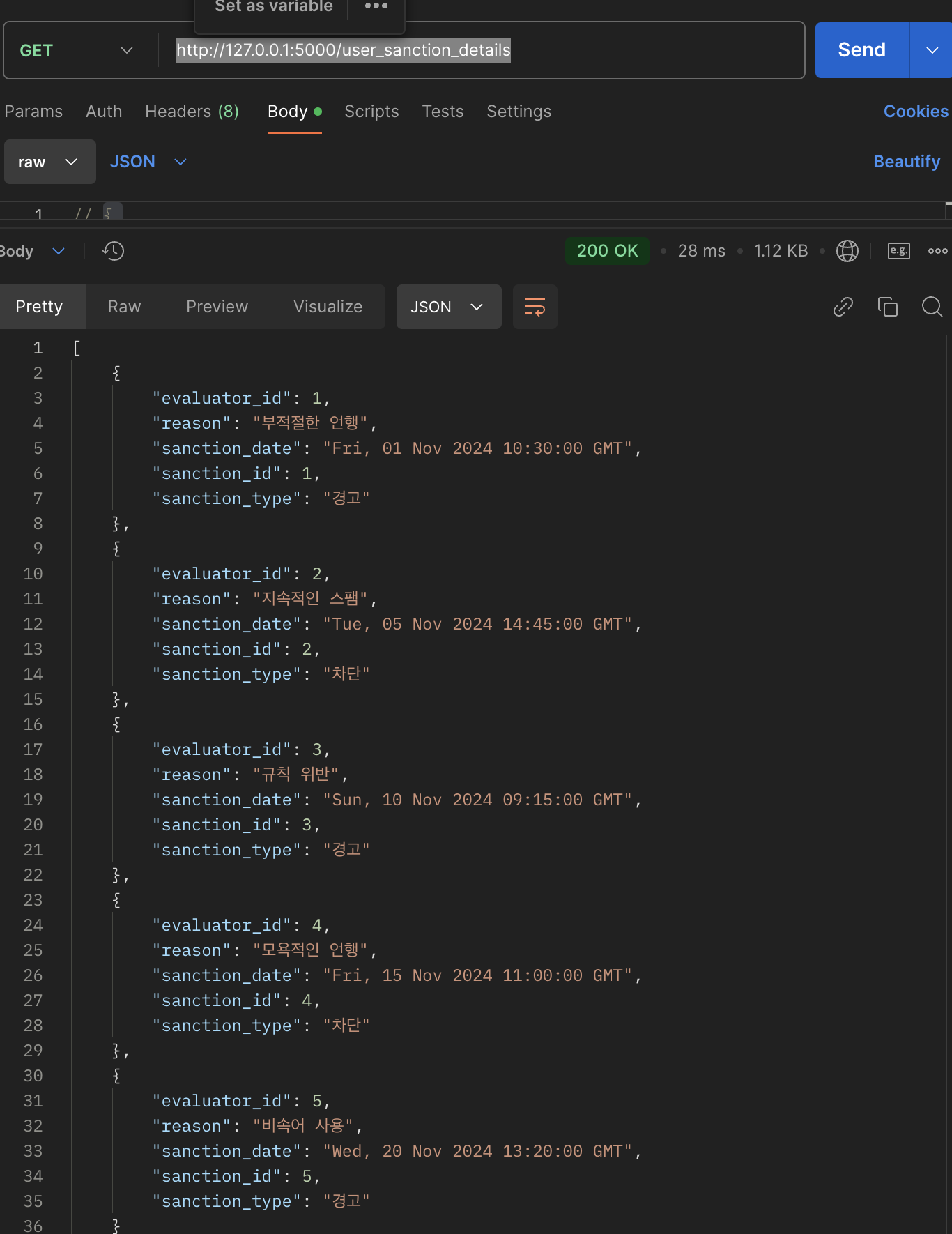
1.[http://127.0.0.1:5000/course\_info[POST]](http://127.0.0.1:5000/course_info*)

2.http://127.0.0.1:5000/course\_evaluation\_info[POST]



3. [http://127.0.0.1:5000/course\_evaluation\_details[GET]](http://127.0.0.1:5000/course_evaluation_details*)

4.[http://127.0.0.1:5000/exam\_info\_details[GET]](http://127.0.0.1:5000/exam_info_details*)

5,[http://127.0.0.1:5000/user\_sanction\_details$ [GET]](http://127.0.0.1:5000/user_sanction_details$*)

테이블에 대한 정보

1. Course

목적: 특정 강의에 대한 기본 정보를 조회합니다.

구성:

course\_id: 강의 ID

course\_name: 강의 이름

professor\_name: 교수 이름

특징: course\_id가 '14466\_006'인 강의에 대한 정보를 제공합니다.

2. CourseEvaluationInfo

목적: 특정 강의에 대한 평가 정보를 집계하여 제공하는 뷰입니다.

구성:

course\_id: 강의 ID

average\_rating: 평균 평점 (소수점 2자리)

total\_evaluations: 총 평가 수

assignment\_amount: 과제 양에 대한 평가 (많음, 보통, 적음)

group\_meeting\_frequency: 조모임 빈도에 대한 평가 (많음, 보통, 적음)

grading\_flexibility: 성적 기준에 대한 평가 (너그러움, 보통, 깐깐함)

특징: course\_id가 '14466\_006'인 강의에 대한 다양한 평가 지표를 제공합니다.

3. CourseEvaluationDetails

목적: 개별 강의평의 세부 정보를 집계하여 제공합니다.

구성:

course\_evaluation\_id: 강의평 ID

average\_rating: 평균 평점

semesters\_attended: 수강한 학기 수

course\_evaluation: 강의평 내용 (여러 내용을 '; '로 구분)

total\_recommend\_count: 총 추천 수

특징: 각 강의평에 대한 집계 정보를 제공하며, 강의평 ID로 그룹화합니다.

4. ExamInfoDetails

목적: 시험 정보에 대한 세부 사항을 집계하여 제공합니다.

구성:

exam\_info\_ids: 시험 정보 ID (중복 제거)

semester: 학기 정보

exam\_round: 시험 회차 (중간고사, 기말고사)

question\_type: 문제 유형 (중복 제거)

exam\_strategy: 시험 전략 (중복 제거)

question\_example: 문제 예시 (중복 제거)

total\_recommend\_count: 총 추천 수

total\_not\_recommend\_count: 총 비추천 수

특징: 학기와 시험 회차로 그룹화하여 시험 관련 정보를 제공합니다.

5. UserSanctionDetails

목적: 사용자 제재에 대한 정보를 제공합니다.

구성:

sanction\_id: 제재 ID

evaluator\_id: 신고를 당한 유저 ID

sanction\_type: 제재 종류

reason: 제재 사유

sanction\_date: 제재 일시

특징: 제재와 관련된 정보를 상세히 제공합니다.