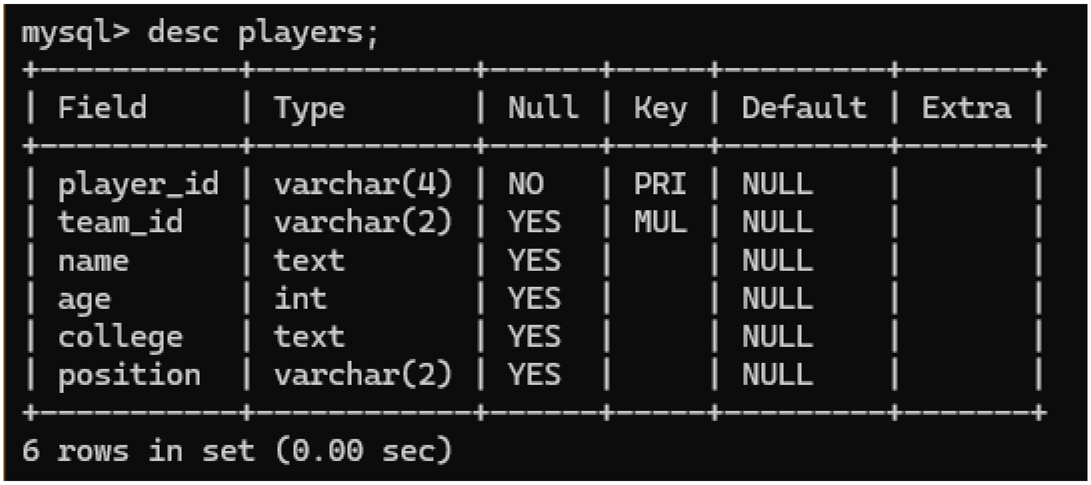
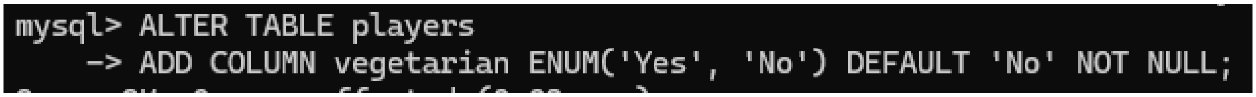
1.a.



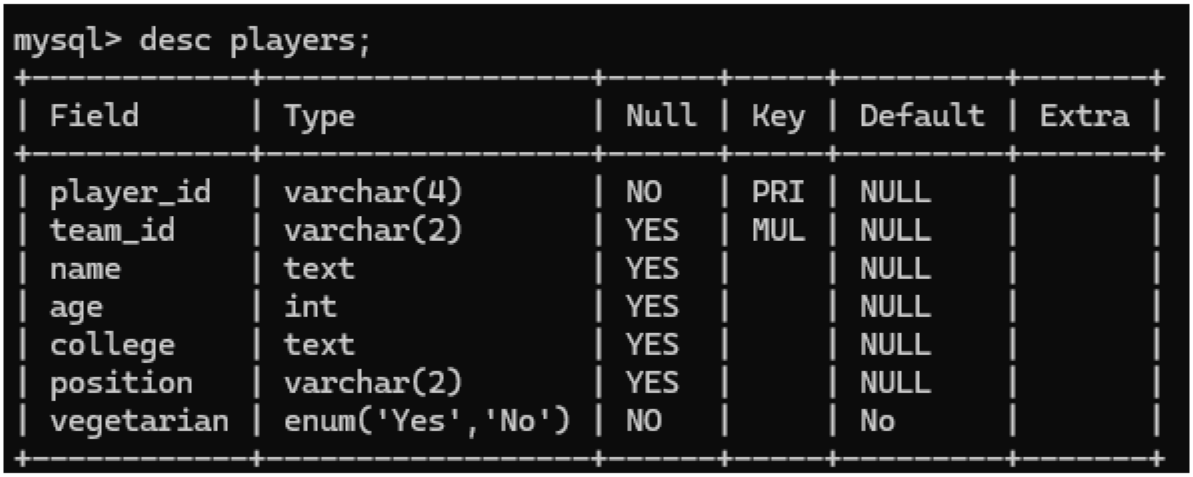
```
mysql> desc players;
+-----------+------------+------+-----+---------+-------+
| Field     | Type       | Null | Key | Default | Extra |
+-----------+------------+------+-----+---------+-------+
| player_id | varchar(4) | NO   | PRI | NULL    |       |
| team_id   | varchar(2) | YES  | MUL | NULL    |       |
| name      | text       | YES  |     | NULL    |       |
| age       | int        | YES  |     | NULL    |       |
| college   | text       | YES  |     | NULL    |       |
| position  | varchar(2) | YES  |     | NULL    |       |
+-----------+------------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```

1. b. and c



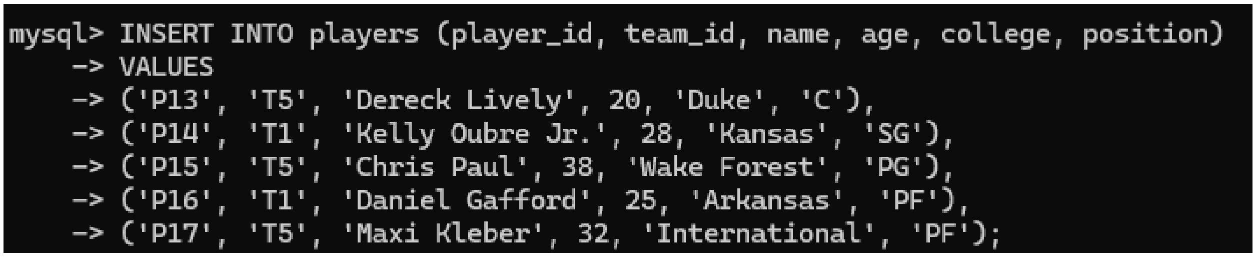
```
mysql> ALTER TABLE players
    -> ADD COLUMN vegetarian ENUM('Yes', 'No') DEFAULT 'No' NOT NULL;
```

1.d.



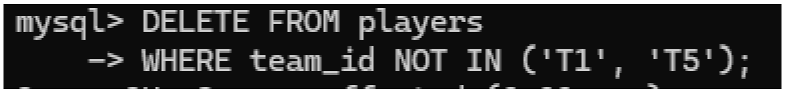
```
mysql> desc players;
+------------+-----------------+------+-----+---------+-------+
| Field      | Type            | Null | Key | Default | Extra |
+------------+-----------------+------+-----+---------+-------+
| player_id  | varchar(4)      | NO   | PRI | NULL    |       |
| team_id    | varchar(2)      | YES  | MUL | NULL    |       |
| name       | text            | YES  |     | NULL    |       |
| age        | int             | YES  |     | NULL    |       |
| college    | text            | YES  |     | NULL    |       |
| position   | varchar(2)      | YES  |     | NULL    |       |
| vegetarian | enum('Yes','No') | NO   |     | No      |       |
+------------+-----------------+------+-----+---------+-------+
```

1.e.



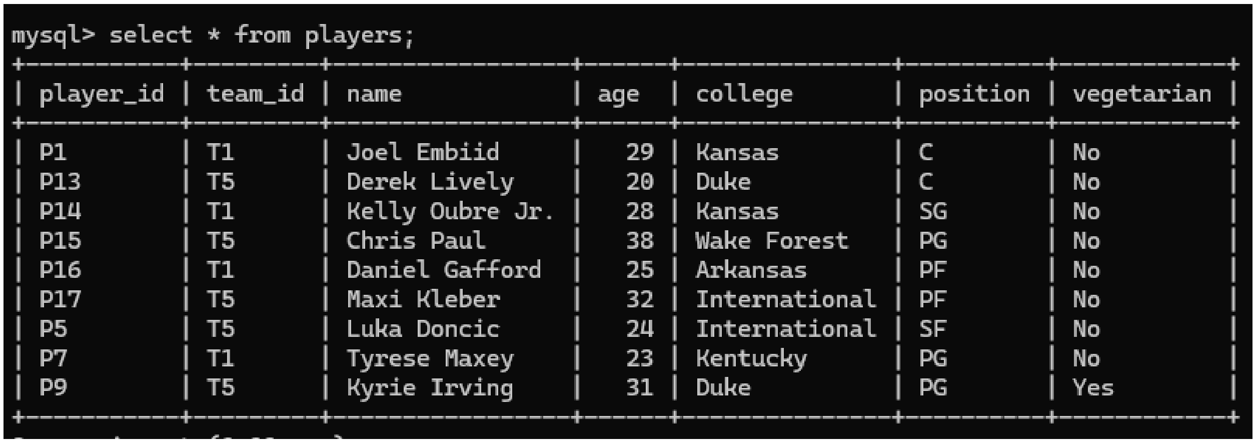
```
mysql> INSERT INTO players (player_id, team_id, name, age, college, position)
    -> VALUES
    -> ('P13', 'T5', 'Dereck Lively', 20, 'Duke', 'C'),
    -> ('P14', 'T1', 'Kelly Oubre Jr.', 28, 'Kansas', 'SG'),
    -> ('P15', 'T5', 'Chris Paul', 38, 'Wake Forest', 'PG'),
    -> ('P16', 'T1', 'Daniel Gafford', 25, 'Arkansas', 'PF'),
    -> ('P17', 'T5', 'Maxi Kleber', 32, 'International', 'PF');
```

1. f



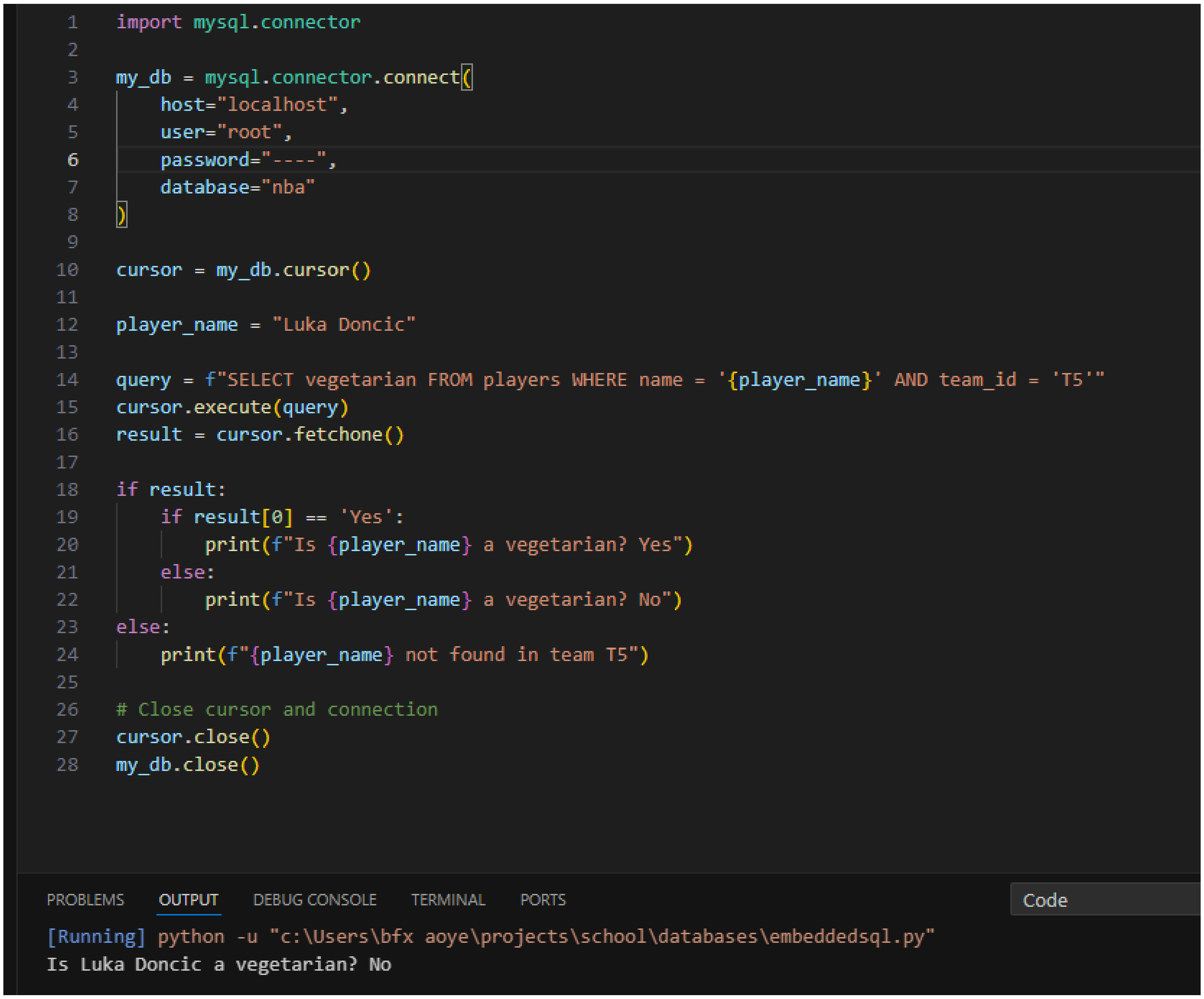
```
mysql> DELETE FROM players
    -> WHERE team_id NOT IN ('T1', 'T5');
```

1. g



```
mysql> select * from players;
+-----------+---------+------------------+------+---------------+----------+------------+
| player_id | team_id | name             | age  | college       | position | vegetarian |
+-----------+---------+------------------+------+---------------+----------+------------+
| P1        | T1      | Joel Embiid      |   29 | Kansas        | C        | No         |
| P13       | T5      | Derek Lively     |   20 | Duke          | C        | No         |
| P14       | T1      | Kelly Oubre Jr.  |   28 | Kansas        | SG       | No         |
| P15       | T5      | Chris Paul       |   38 | Wake Forest   | PG       | No         |
| P16       | T1      | Daniel Gafford   |   25 | Arkansas      | PF       | No         |
| P17       | T5      | Maxi Kleber      |   32 | International | PF       | No         |
| P5        | T5      | Luka Doncic      |   24 | International | SF       | No         |
| P7        | T1      | Tyrese Maxey     |   23 | Kentucky      | PG       | No         |
| P9        | T5      | Kyrie Irving     |   31 | Duke          | PG       | Yes        |
+-----------+---------+------------------+------+---------------+----------+------------+
```

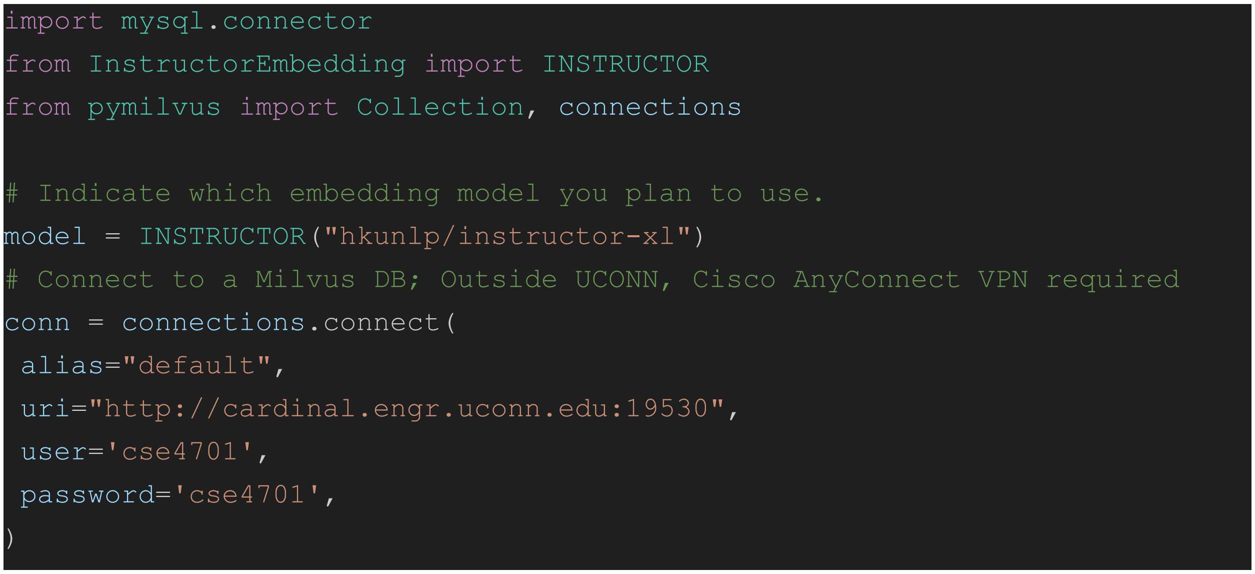
2. A.



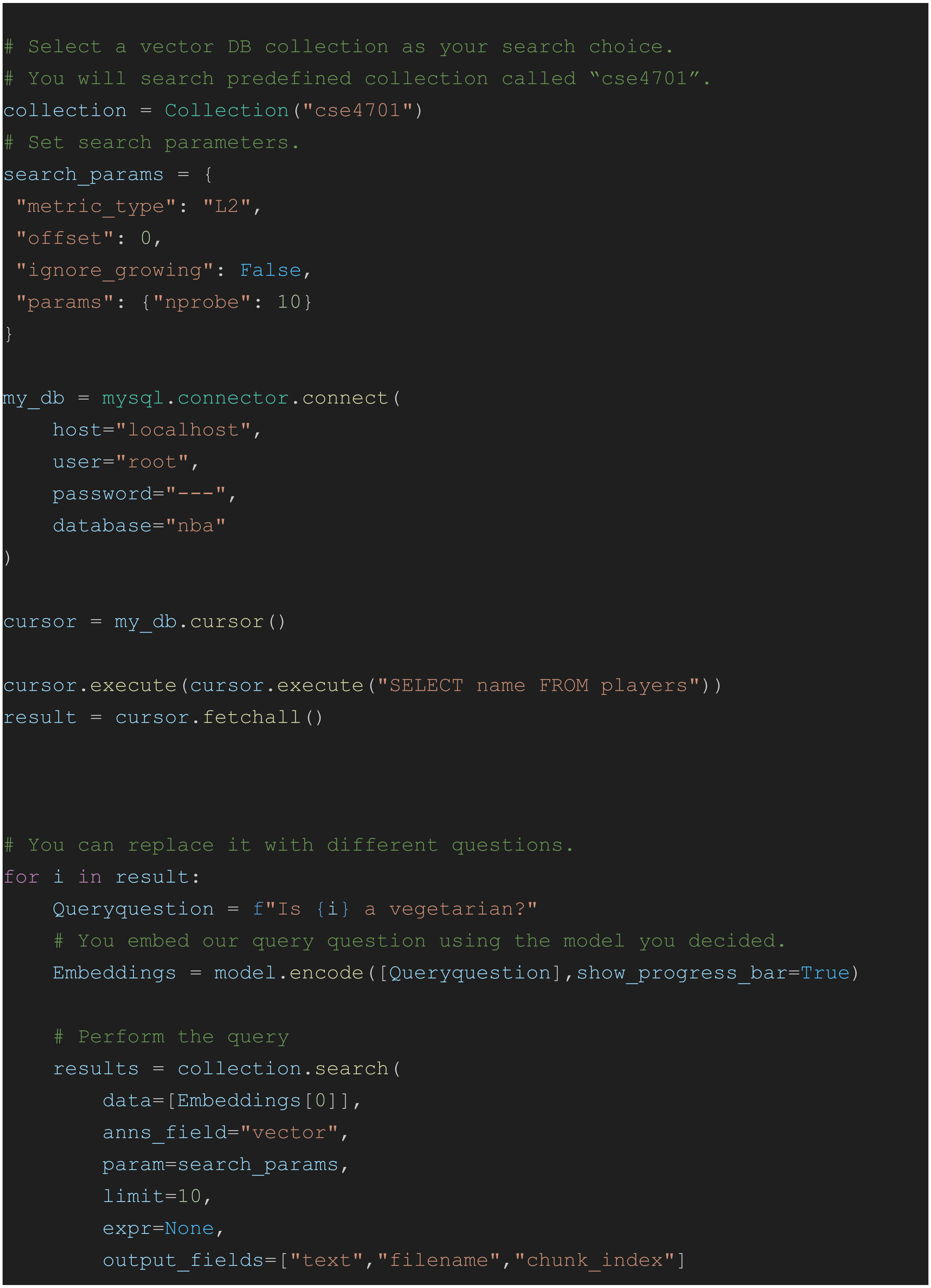
```
import mysql.connector

my_db = mysql.connector.connect(
    host="localhost",
    user="root",
    password="----",
    database="nba"
)

cursor = my_db.cursor()

player_name = "Luka Doncic"

query = f"SELECT vegetarian FROM players WHERE name = '{player_name}' AND team_id = 'T5'"
cursor.execute(query)
result = cursor.fetchone()

if result:
    if result[0] == 'Yes':
        print(f"Is {player_name} a vegetarian? Yes")
    else:
        print(f"Is {player_name} a vegetarian? No")
else:
    print(f"{player_name} not found in team T5")

# Close cursor and connection
cursor.close()
my_db.close()
```

```
[Running] python -u "c:\Users\bfx aoye\projects\school\databases\embeddedsql.py"
Is Luka Doncic a vegetarian? No
```

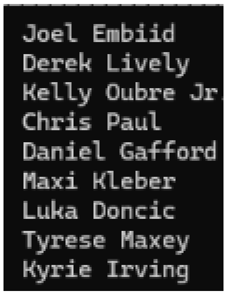
2. B.



```
import mysql.connector
from InstructorEmbedding import INSTRUCTOR
from pymilvus import Collection, connections

# Indicate which embedding model you plan to use.
model = INSTRUCTOR("hkunlp/instructor-xl")
# Connect to a Milvus DB; Outside UCONN, Cisco AnyConnect VPN required
conn = connections.connect(
 alias="default",
 uri="http://cardinal.engr.uconn.edu:19530",
 user='cse4701',
 password='cse4701',
)
```



```python
# Select a vector DB collection as your search choice.
# You will search predefined collection called “cse4701”.
collection = Collection("cse4701")
# Set search parameters.
search_params = {
 "metric_type": "L2",
 "offset": 0,
 "ignore_growing": False,
 "params": {"nprobe": 10}
}

my_db = mysql.connector.connect(
    host="localhost",
    user="root",
    password="---",
    database="nba"
)

cursor = my_db.cursor()

cursor.execute(cursor.execute("SELECT name FROM players"))
result = cursor.fetchall()



# You can replace it with different questions.
for i in result:
    Queryquestion = f"Is {i} a vegetarian?"
    # You embed our query question using the model you decided.
    Embeddings = model.encode([Queryquestion],show_progress_bar=True)

    # Perform the query
    results = collection.search(
        data=[Embeddings[0]],
        anns_field="vector",
        param=search_params,
        limit=10,
        expr=None,
        output_fields=["text","filename","chunk_index"]
        )
        # Print out query results
    for hit in results[0]:
        print("Vector distance:", hit.distance)
        print("Text: ",hit.entity.get('text'))
        print("Filename: ",hit.entity.get('filename'))
        print("Chunk index: ",hit.entity.get('chunk_index'))

# Close cursor and connection
cursor.close()
my_db.close()
```

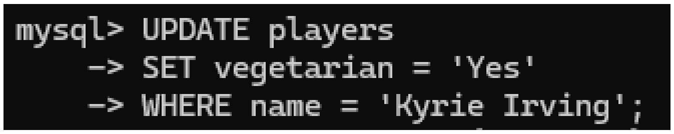


```
Joel Embiid
Derek Lively
Kelly Oubre Jr
Chris Paul
Daniel Gafford
Maxi Kleber
Luka Doncic
Tyrese Maxey
Kyrie Irving
```

We don’t know if Maxi Kleber is a vegetarian
Text: We do not know if Maxi is a vegetarian or not. Vegetarians don't eat any food products made from
Filename: Maxi_Kleber.txt
Chunk index: 12

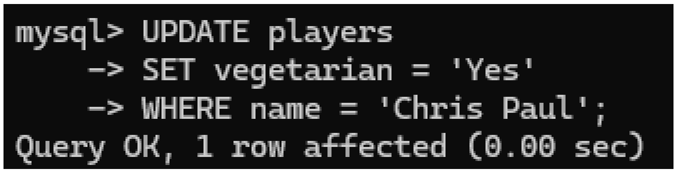
Chris Paul is a vegetarian
Text: Basketball player Chris Paul switched to a vegan diet in 2019 at the age of 34 when he was with the
Filename: Chris_Paul.txt
Chunk index: 1

Kyrie Irving is a vegetarian
Text: Kyrie Irving switched to a strict vegan lifestyle in 2017 when he was with the Boston Celtics after
Filename: Kyrie_Irving .txt
Chunk index: 3

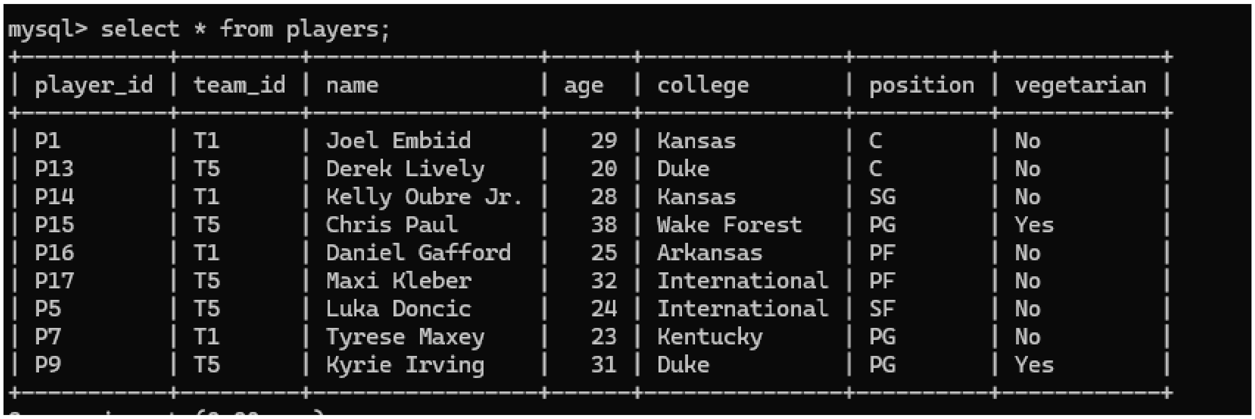


```
mysql> UPDATE players
    -> SET vegetarian = 'Yes'
    -> WHERE name = 'Kyrie Irving';
```

2. C.

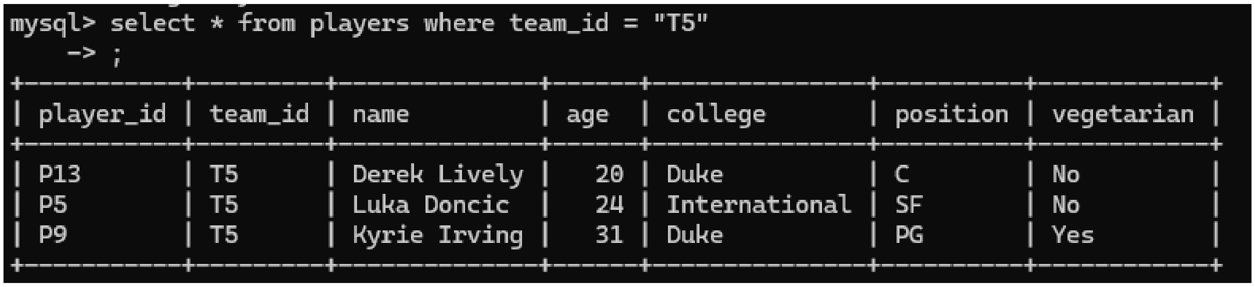


```
mysql> UPDATE players
    -> SET vegetarian = 'Yes'
    -> WHERE name = 'Chris Paul';
Query OK, 1 row affected (0.00 sec)
```



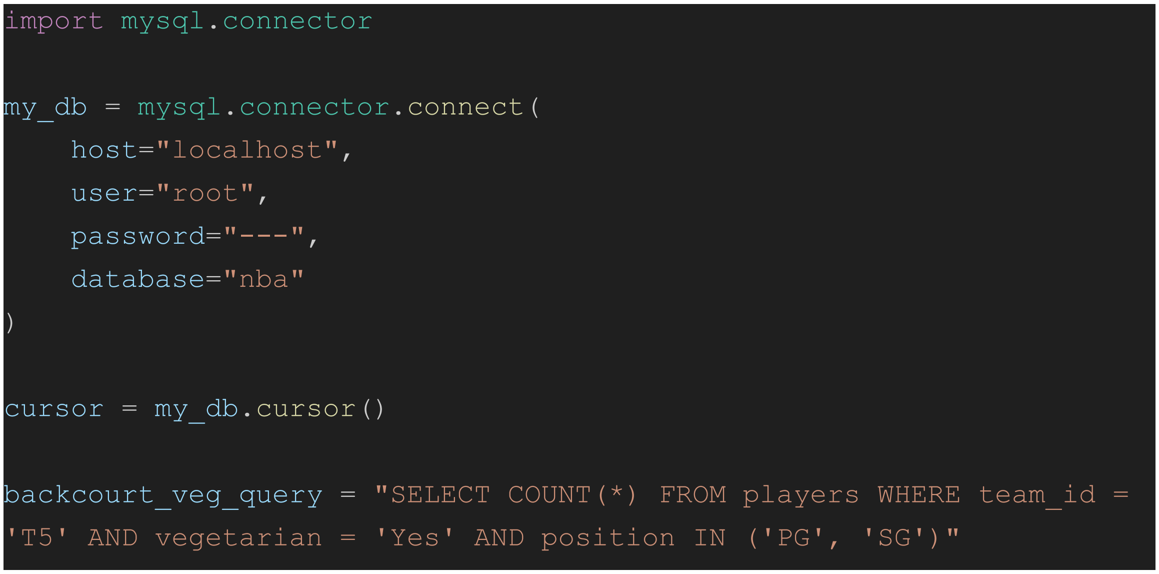
```
mysql> select * from players;
+-----------+---------+-------------------+------+---------------+----------+------------+
| player_id | team_id | name              | age  | college       | position | vegetarian |
+-----------+---------+-------------------+------+---------------+----------+------------+
| P1        | T1      | Joel Embiid       |   29 | Kansas        | C        | No         |
| P13       | T5      | Derek Lively      |   20 | Duke          | C        | No         |
| P14       | T1      | Kelly Oubre Jr.   |   28 | Kansas        | SG       | No         |
| P15       | T5      | Chris Paul        |   38 | Wake Forest   | PG       | Yes        |
| P16       | T1      | Daniel Gafford    |   25 | Arkansas      | PF       | No         |
| P17       | T5      | Maxi Kleber       |   32 | International | PF       | No         |
| P5        | T5      | Luka Doncic       |   24 | International | SF       | No         |
| P7        | T1      | Tyrese Maxey      |   23 | Kentucky      | PG       | No         |
| P9        | T5      | Kyrie Irving      |   31 | Duke          | PG       | Yes        |
+-----------+---------+-------------------+------+---------------+----------+------------+
```

2.e.

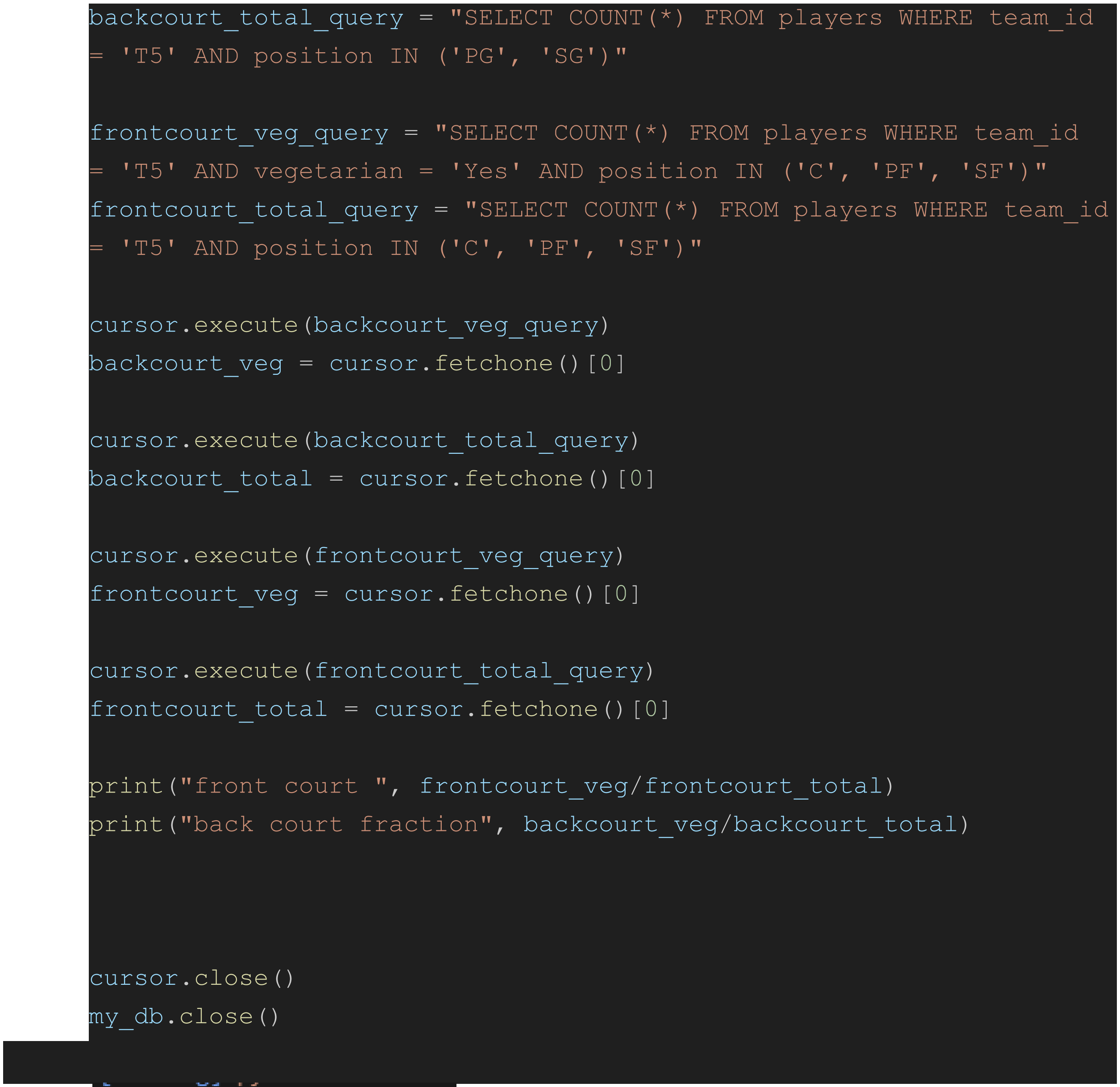


```
mysql> select * from players where team_id = "T5"
    -> ;
+-----------+---------+--------------+------+---------------+----------+------------+
| player_id | team_id | name         | age  | college       | position | vegetarian |
+-----------+---------+--------------+------+---------------+----------+------------+
| P13       | T5      | Derek Lively |   20 | Duke          | C        | No         |
| P5        | T5      | Luka Doncic  |   24 | International | SF       | No         |
| P9        | T5      | Kyrie Irving |   31 | Duke          | PG       | Yes        |
+-----------+---------+--------------+------+---------------+----------+------------+
```

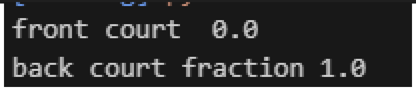
3.



```
import mysql.connector

my_db = mysql.connector.connect(
    host="localhost",
    user="root",
    password="---",
    database="nba"
)

cursor = my_db.cursor()

backcourt_veg_query = "SELECT COUNT(*) FROM players WHERE team_id =
'T5' AND vegetarian = 'Yes' AND position IN ('PG', 'SG')"
```



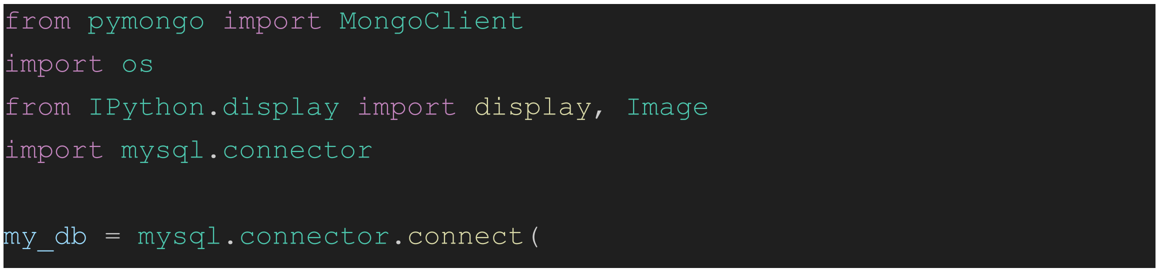
```
backcourt_total_query = "SELECT COUNT(*) FROM players WHERE team_id
= 'T5' AND position IN ('PG', 'SG')"

frontcourt_veg_query = "SELECT COUNT(*) FROM players WHERE team_id
= 'T5' AND vegetarian = 'Yes' AND position IN ('C', 'PF', 'SF')"
frontcourt_total_query = "SELECT COUNT(*) FROM players WHERE team_id
= 'T5' AND position IN ('C', 'PF', 'SF')"

cursor.execute(backcourt_veg_query)
backcourt_veg = cursor.fetchone()[0]

cursor.execute(backcourt_total_query)
backcourt_total = cursor.fetchone()[0]

cursor.execute(frontcourt_veg_query)
frontcourt_veg = cursor.fetchone()[0]

cursor.execute(frontcourt_total_query)
frontcourt_total = cursor.fetchone()[0]

print("front court ", frontcourt_veg/frontcourt_total)
print("back court fraction", backcourt_veg/backcourt_total)




cursor.close()
my_db.close()
```



```
front court  0.0
back court fraction 1.0
```

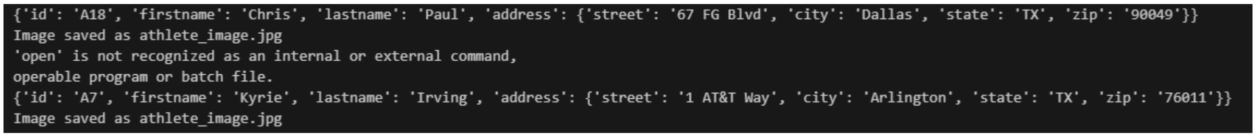
So, no front court players were vegetarian, but all back court players were vegetarian.

4.



```
from pymongo import MongoClient
import os
from IPython.display import display, Image
import mysql.connector

my_db = mysql.connector.connect(
    host="localhost",
    user="root",
    password="---",
    database="nba"
)
cursor = my_db.cursor()
query = "SELECT name FROM players WHERE vegetarian = 'Yes'"
cursor.execute(query)
athletes = cursor.fetchall()
client =
MongoClient("mongodb://cse4701:cse4701@137.99.245.21:27017/myDatabas
e")
db = client['myDatabase']
athletes_collection = db['cse4701_ext']
for i in athletes:
    athlete_name = i[0].split()[0]
    document =
athletes_collection.find_one({'description.firstname':
athlete_name})
    print(document['description'])
    if document:
        image_data = document.get('images')
        # If image data exists, save it to a file
        if image_data:
            image_file_path = 'athlete_image.jpg'
            with open(image_file_path, 'wb') as image_file:
                image_file.write(image_data)
                print(f"Image saved as {image_file_path}")

    # Automatically open the image file with the default application
            os.system(f'open {image_file_path}')
        else:
            print(f"No image found for {athlete_name}")

    else:
        print(f"No document found for athlete with name
{athlete_name}")


cursor.close()
```

```
my_db.close()
```



```
{'id': 'A18', 'firstname': 'Chris', 'lastname': 'Paul', 'address': {'street': '67 FG Blvd', 'city': 'Dallas', 'state': 'TX', 'zip': '90049'}}
Image saved as athlete_image.jpg
'open' is not recognized as an internal or external command,
operable program or batch file.
{'id': 'A7', 'firstname': 'Kyrie', 'lastname': 'Irving', 'address': {'street': '1 AT&T Way', 'city': 'Arlington', 'state': 'TX', 'zip': '76011'}}
Image saved as athlete_image.jpg
```

5. The easiest way to do this would be to fine tune and ensemble several pre-trained transformers to specifically identify if sentences are describing if an individual is vegetarian and constrain their answer to return a yes or no answer to “if someone is vegetarian”. This is great because generally, it will be fast, and have the added benefit of ensembling which can reduce made up behaviors. Obviously, it may or may not perform as well as a person, but it most likely will be able to do this in bulk with fairly decent accuracy. Fine tuning transformer models would take far too long, but it would just be standard ensembling methods for voting on outcomes.

```
num_classes = 2

finetuned_classes = [
    ‘yes’,
    ‘no’
]

!python main.py \
  --dataset_file "custom" \
  --path "/content/data/custom/" \
  --output_dir "outputs" \
  --resume "detr-r50_no-class-head.pth" \
  --num_classes $num_classes \
  --epochs 10
```