Description of the application:

The menu allows the user to select several options: Show the games with the most global score for a specific console, show the ratings for the games in a saga, show how many times games have been played in a console, show all games, in every console, with a specific global score, add a new game and to all the required tables, and finally, exit the application.

Code

```
import mysql.connector

""" Description of the application:
The menu allows the user to select several options: Show the games with the
most global score
for a specific console, show the ratings for the games in a saga, show how
many times games
have been played in a console, show all games, in every console, with a
specific global score,
add a new game and to all the required tables, and finally, exit the
application.  """

config = {
  'user':'test',    # could be root, or a user you created, I created
'testuser'
  'password':'none',  # the password for that use
  'database':'games',   # the database to connect to
  'host':'192.168.1.128',   # localhost
  'allow_local_infile':True  # needed so can load local files
  }

def print_menu():
  print("\n"*10)
  print("Choose the action to take")
  print("1. Show a number of top games from a game system")
  print("2. Show game ratings of game collections")
  print("3. Show times played in a game system")
  print("4. Show all games with a specific rating")
  print("5. Add a new game")
  print("6. Exit")
```

```
if __name__ == "__main__":
  mydb = mysql.connector.connect(**config)
  myc = mydb.cursor()
  while True:
    query = ""
    print_menu()
    x = int(input("Select your option\n"))
    match x:
      case 1:
        num_games = int(input("How many games do you want to see?\n"))
        game_system = str(input("From which system do you want to see the
games?\n"))
        query = """select distinct g.title, e.global_score from games g,
game_system gs,
        availability a, partof po, enjoyment e where gs.sName = a.SystemName
and
        g.title = po.title and e.game = g.title and gs.sName = \"{}\"
        order by e.global_score desc limit
{}""".format(game_system,num_games)
      case 2:
        saga_like = str(input("Wildcard for the sagas to show\n"))
        query = """with games as (select s.members from saga s where s.saga
        like \"%{}%\") select g.members, e.global_score from games g,
enjoyment e
        where g.members = e.game""".format(saga_like)
      case 3:
        game_system = str(input("From which system do you want to see the
games?\n"))
        query = """with playedtimes as (select distinct g.title,
p.played_times from games g,
        game_system gs, availability a, partof po, enjoyment e, played p
        where gs.sName = a.SystemName and g.title = po.title and e.game =
g.title
        and e.game = p.title and gs.sName = \"{}\") select sum(played_times)
from playedtimes""".format(game_system)
      case 4:
        expected_score = int(input("What is the exact score to show the
games?\n"))
        query = """select game from enjoyment where global_score = {}
        """.format(expected_score)
      case 5:
        title = str(input("Title: \n"))
        systemname = str(input("System in which you can play it: \n"))
        music_score = int(input("Music Score: \n"))
```

```
        graphics_score = int(input("Graphics Score: \n"))
        gameplay_score = int(input("Gameplay Score: \n"))
        story_score = int(input("Story Score: \n"))
        global_score = (music_score + graphics_score + gameplay_score +
story_score)/4
        release_date = str(input("Release Date: \n"))
        developer = str(input("Developer: \n"))
        classification = str(input("Classification: \n"))
        Genre = str(input("Genre: \n"))
        saga = str(input("Saga: \n"))
        played_times = int(input("Played items: \n"))
        try:
          insert_statement = "insert into games values
(\"{}\",\"{}\",\"{}\",\"{}\",\"{}\")".format(title, release_date, developer,
classification, Genre)
          myc.execute(insert_statement)
        except Exception as e:
          print(e)
          print("That game appears to be already in the database")
        try:

          insert_statement = "insert into saga values
(\"{}\",1,\"{}\")".format(saga, title)
          myc.execute(insert_statement)
        except Exception as e:
          print(e)
          print("Either the saga already exists, or the game does not
exist")
        try:
          insert_statement = "insert into availability values
(\"{}\",\"{}\")".format(systemname, saga)
          myc.execute(insert_statement)
          insert_statement = "insert into enjoyment values
(\"{}\",{},{},{},{},{})".format(title, music_score, graphics_score,
gameplay_score, story_score, global_score)
          myc.execute(insert_statement)
          insert_statement = "insert into ownedon values
(\"{}\",\"{}\")".format(title, systemname)
          myc.execute(insert_statement)
          insert_statement = "insert into partof values
(\"{}\",\"{}\")".format(saga, title)
          myc.execute(insert_statement)
          query = "select * from games where title = \"{}\"".format(title)
        except Exception as e:
          print(e)
```

```
        mydb.commit()
      case 6:
        break
      case _:
        pass
    print(query)
    myc.execute(query)
    for x in myc:
      print(x)
  mydb.close()
```

Results

Choose the action to take

1. Show a number of top games from a game system

2. Show game ratings of game collections

3. Show times played in a game system

4. Show all games with a specific rating

5. Add a new game

6. Exit

Select your option

5

Title:

Test

System in which you can play it:

PlayStation 3

Music Score:

10

Graphics Score:

10

Gameplay Score:

10

Story Score:

10

Release Date:

2020-12-12

Developer:

Santa Monica Studio

Classification:

M

Genre:

Action

Saga:

GGG

Played items:

17

select * from games where title = "Test"

('Test', datetime.date(2020, 12, 12), 'Santa Monica Studio', 'M', 'Action')

Choose the action to take

1. Show a number of top games from a game system

2. Show game ratings of game collections

3. Show times played in a game system

4. Show all games with a specific rating

5. Add a new game

6. Exit

Select your option

4

What is the exact score to show the games?

10

select game from enjoyment where global_score = 10

('ActRaiser: Electric Boogaloo',)

('Cannon Fodder: Isolation',)

('Test',)

Choose the action to take

1. Show a number of top games from a game system

2. Show game ratings of game collections

3. Show times played in a game system

4. Show all games with a specific rating

5. Add a new game

6. Exit

Select your option

1

How many games do you want to see?

10

From which system do you want to see the games?

Wii

```
select distinct g.title, e.global_score from games g, game_system gs,
    availability a, partof po, enjoyment e where gs.sName = a.SystemName and
    g.title = po.title and e.game = g.title and gs.sName = "Wii"
    order by e.global_score desc limit 10
```

('Test', 10)

('ActRaiser: Electric Boogaloo', 10)

('Cannon Fodder: Isolation', 10)

('Ace Combat: 3', 9)

('Anno: Heroes', 9)

('Ar Tonelico: Tokyo Drift', 9)

('Alien Syndrome: yeah wow', 9)

('Alex Kidd: Zero Dawn', 9)

('Anno: im just bored', 9)

('Aleste: Ascension', 9)

Choose the action to take

1. Show a number of top games from a game system

2. Show game ratings of game collections

3. Show times played in a game system

4. Show all games with a specific rating

5. Add a new game

6. Exit

Select your option

2

Wildcard for the sagas to show

shock

```
with games as (select s.members from saga s where s.saga
    like "%shock%") select g.members, e.global_score from games g, enjoyment e
    where g.members = e.game
```

('BioShock: 1942', 6)

('BioShock: 2', 7)

('BioShock: 2 Fast 2 Furious', 4)

('BioShock: 3', 5)

('BioShock: 4', 6)

('BioShock: 5', 4)

('BioShock: 6', 9)

('BioShock: Advanced Warfare', 6)

('BioShock: Armored Fury', 5)

('BioShock: Ascension', 5)

('BioShock: Back to Karkand', 4)

('BioShock: Bad Company', 4)

('BioShock: Black Flag', 6)

('BioShock: Black Ops', 6)

('BioShock: Brotherhood', 6)

('BioShock: cats?', 6)

('BioShock: Chains of Olympus', 5)

('BioShock: Electric Boogaloo', 3)

('BioShock: Forbidden West', 6)

('BioShock: Ghost of Sparta', 3)

('BioShock: Ghosts', 3)

('BioShock: Hardline', 4)

('BioShock: Heroes', 5)

('BioShock: I', 5)

('BioShock: II', 3)

('BioShock: III', 6)

('BioShock: im just bored', 4)

('BioShock: Infinity', 7)

('BioShock: Isolation', 4)

('BioShock: Madness Returns', 5)

('BioShock: Modern Combat', 4)

('BioShock: Modern Warfare', 4)

('BioShock: more games', 7)

('BioShock: more garbage data', 4)

('BioShock: Odyssey', 5)

('BioShock: Origins', 7)

('BioShock: Revelations', 4)

('BioShock: Rogue', 6)

('BioShock: Syndicate', 4)

('BioShock: Tokyo Drift', 5)

('BioShock: Valhalla', 5)

('BioShock: Vietnam', 7)

('BioShock: World at War', 4)

('BioShock: yeah wow', 3)

('BioShock: Zero Dawn', 5)

('System Shock: 1942', 6)

('System Shock: 2', 5)

('System Shock: 2 Fast 2 Furious', 5)

('System Shock: 3', 6)

('System Shock: 4', 5)

('System Shock: 5', 5)

('System Shock: 6', 3)

('System Shock: Advanced Warfare', 5)

('System Shock: Armored Fury', 3)

('System Shock: Ascension', 5)

('System Shock: Back to Karkand', 4)

('System Shock: Bad Company', 4)

('System Shock: Black Flag', 6)

('System Shock: Black Ops', 3)

('System Shock: Brotherhood', 5)

('System Shock: cats?', 4)

('System Shock: Chains of Olympus', 7)

('System Shock: Electric Boogaloo', 3)

('System Shock: Forbidden West', 7)

('System Shock: Ghost of Sparta', 5)

('System Shock: Ghosts', 2)

('System Shock: Hardline', 6)

('System Shock: Heroes', 6)

('System Shock: I', 4)

('System Shock: II', 6)

('System Shock: III', 7)

('System Shock: im just bored', 7)

('System Shock: Infinity', 6)

('System Shock: Isolation', 3)

('System Shock: Madness Returns', 6)

('System Shock: Modern Combat', 5)

('System Shock: Modern Warfare', 7)

('System Shock: more games', 5)

('System Shock: more garbage data', 6)

('System Shock: Odyssey', 3)

('System Shock: Origins', 6)

('System Shock: Revelations', 7)

('System Shock: Rogue', 7)

('System Shock: Syndicate', 1)

('System Shock: Tokyo Drift', 6)

('System Shock: Valhalla', 6)

('System Shock: Vietnam', 8)

('System Shock: World at War', 6)

('System Shock: yeah wow', 4)

('System Shock: Zero Dawn', 8)

Choose the action to take

1. Show a number of top games from a game system

2. Show game ratings of game collections

3. Show times played in a game system

4. Show all games with a specific rating

5. Add a new game

6. Exit

Select your option

3

From which system do you want to see the games?

Wii

with playedtimes as (select distinct g.title, p.played_times from games g,

game_system gs, availability a, partof po, enjoyment e, played p

where gs.sName = a.SystemName and g.title = po.title and e.game = g.title

and e.game = p.title and gs.sName = "Wii") select sum(played_times) from playedtimes

(Decimal('440610'),)

Choose the action to take

1. Show a number of top games from a game system

2. Show game ratings of game collections

3. Show times played in a game system

4. Show all games with a specific rating

5. Add a new game

6. Exit

Select your option

4

What is the exact score to show the games?

10

select game from enjoyment where global_score = 10

('ActRaiser: Electric Boogaloo',)

('Cannon Fodder: Isolation',)

('Test',)

Choose the action to take

1. Show a number of top games from a game system

2. Show game ratings of game collections

3. Show times played in a game system

4. Show all games with a specific rating

5. Add a new game

6. Exit

Select your option