**1ST SET OF STATEMENTS:**

**#List all competitions along with their category name**

mycursor.execute("""SELECT c.competition\_name, cat.category\_name

FROM Competition c

JOIN Category cat ON c.category\_id = cat.category\_id

""" )

out=mycursor.fetchall()

print(tabulate(out, headers=[i[0] for i in mycursor.description], tablefmt='grid'))

**#Count the number of competitions in each category**

mycursor.execute("""SELECT cat.category\_name, COUNT(c.competition\_id) AS num\_competitions

FROM Competition c

JOIN Category cat ON c.category\_id = cat.category\_id

GROUP BY cat.category\_name

""" )

out=mycursor.fetchall()

print(tabulate(out, headers=[i[0] for i in mycursor.description], tablefmt='grid'))

**#Find all competitions of type 'doubles'**

mycursor.execute("""

SELECT competition\_name, type

FROM Competition

WHERE type = 'doubles';

""" )

out=mycursor.fetchall()

print(tabulate(out, headers=[i[0] for i in mycursor.description], tablefmt='grid'))

**#Get competitions that belong to a specific category (e.g., ITF Men)**

mycursor.execute("""

SELECT competition\_name, cat.category\_name

FROM Competition c

JOIN Category cat ON c.category\_id = cat.category\_id

WHERE cat.category\_name = 'ITF Men'

""" )

out=mycursor.fetchall()

print(tabulate(out, headers=[i[0] for i in mycursor.description], tablefmt='grid'))

**#Identify parent competitions and their sub-competitions**

mycursor.execute("""

SELECT competition\_id, competition\_name, parent\_id

FROM Competition

WHERE parent\_id IS NOT NULL;

""")

out = mycursor.fetchall()

print(tabulate(out, headers=[i[0] for i in mycursor.description], tablefmt='grid'))

**#Analyze the distribution of competition types by category**

mycursor.execute("""

SELECT cat.category\_name, c.type, COUNT(c.competition\_id) AS num\_competitions

FROM Competition c

JOIN Category cat ON c.category\_id = cat.category\_id

GROUP BY cat.category\_name, c.type

""" )

out=mycursor.fetchall()

print(tabulate(out, headers=[i[0] for i in mycursor.description], tablefmt='grid'))

**#List all competitions with no parent (top-level competitions)**

mycursor.execute("""

SELECT competition\_name

FROM Competition

WHERE parent\_id IS NULL

""" )

out=mycursor.fetchall()

print(tabulate(out, headers=[i[0] for i in mycursor.description], tablefmt='grid'))

**2ND SET OF STATEMENTS:**

**#List all venues along with their associated complex name**

mycursor.execute("""

SELECT v.venue\_name, c.complex\_name

FROM venues v

JOIN complexes c ON v.complex\_id = c.complex\_id

""" )

out=mycursor.fetchall()

print(tabulate(out, headers=[i[0] for i in mycursor.description], tablefmt='grid'))

**#Count the number of venues in each complex**

mycursor.execute("""

SELECT c.complex\_name, COUNT(v.venue\_id) AS venue\_count

FROM complexes c

JOIN venues v ON c.complex\_id = v.complex\_id

GROUP BY c.complex\_name

""" )

out=mycursor.fetchall()

print(tabulate(out, headers=[i[0] for i in mycursor.description], tablefmt='grid'))

**#Get details of venues in a specific country (e.g., Chile)**

mycursor.execute("""

SELECT \* FROM venues

WHERE country\_name = 'Chile'

""" )

out=mycursor.fetchall()

print(tabulate(out, headers=[i[0] for i in mycursor.description], tablefmt='grid'))

**#Identify all venues and their timezones**

mycursor.execute( """

SELECT venue\_name, timezone FROM venues

""" )

out=mycursor.fetchall()

print(tabulate(out, headers=[i[0] for i in mycursor.description], tablefmt='grid'))

**#Find complexes that have more than one venue**

mycursor.execute( """

SELECT c.complex\_name, COUNT(v.venue\_id) AS venue\_count

FROM complexes c

JOIN venues v ON c.complex\_id = v.complex\_id

GROUP BY c.complex\_id

HAVING COUNT(v.venue\_id) > 1

""" )

out=mycursor.fetchall()

print(tabulate(out, headers=[i[0] for i in mycursor.description], tablefmt='grid'))

**#List venues grouped by country**

mycursor.execute("""

SELECT country\_name, GROUP\_CONCAT(venue\_name) AS venues

FROM venues

GROUP BY country\_name

""")

out=mycursor.fetchall()

print(tabulate(out, headers=[i[0] for i in mycursor.description], tablefmt='grid'))

**#Find all venues for a specific complex (e.g., Nacional)**

mycursor.execute("""

SELECT v.\*

FROM venues v

JOIN complexes c ON v.complex\_id = c.complex\_id

WHERE c.complex\_name = 'Nacional'

""")

out=mycursor.fetchall()

print(tabulate(out, headers=[i[0] for i in mycursor.description], tablefmt='grid'))

**3RD SET OF STATEMENTS:**

**# Get all competitors with their rank and points**

mycursor.execute("""

SELECT c.name, cr.rank, cr.points

FROM competitors c

JOIN competitor\_rankings cr ON c.id = cr.competitor\_id

""")

out=mycursor.fetchall()

print(tabulate(out, headers=[i[0] for i in mycursor.description], tablefmt='grid'))

**#Find competitors ranked in the top 5**

mycursor.execute("""

SELECT c.name, cr.rank, cr.points

FROM competitors c

JOIN competitor\_rankings cr ON c.id = cr.competitor\_id

WHERE cr.rank <= 5

ORDER BY cr.rank

""")

out=mycursor.fetchall()

print(tabulate(out, headers=[i[0] for i in mycursor.description], tablefmt='grid'))

**#List competitors with no rank movement (stable rank)**

mycursor.execute("""

SELECT c.name, cr.rank, cr.movement

FROM competitors c

JOIN competitor\_rankings cr ON c.id = cr.competitor\_id

WHERE cr.movement = 0

""")

out=mycursor.fetchall()

print(tabulate(out, headers=[i[0] for i in mycursor.description], tablefmt='grid'))

**#Get total points of competitors from Croatia**

mycursor.execute("""

SELECT SUM(cr.points) AS total\_points

FROM competitors c

JOIN competitor\_rankings cr ON c.id = cr.competitor\_id

WHERE c.country = 'Croatia'

""")

out=mycursor.fetchall()

print(tabulate(out, headers=[i[0] for i in mycursor.description], tablefmt='grid'))

**#Count number of competitors per country**

mycursor.execute("""

SELECT country, COUNT(\*) AS competitor\_count

FROM competitors

GROUP BY country

ORDER BY competitor\_count DESC

""")

out=mycursor.fetchall()

print(tabulate(out, headers=[i[0] for i in mycursor.description], tablefmt='grid'))

**#Find competitor with the highest points**

mycursor.execute("""

SELECT c.name, cr.points

FROM competitors c

JOIN competitor\_rankings cr ON c.id = cr.competitor\_id

ORDER BY cr.points DESC

LIMIT 1

""")

out=mycursor.fetchall()

print(tabulate(out, headers=[i[0] for i in mycursor.description], tablefmt='grid'))