For this assignment you will be required to write SQL queries to answer to complete the following tasks. Please use the submission boxes provided to record your answers. An example is given below.

Example	
Task	Return the id and name of all athletes.
Explanation	This query should return a table with two columns, one for the id and one for the name of the athletes.

SQL Solution	`SELECT AthleteID, AthleteName`
	`FROM Athletes`
	`LIMIT 10;`

Output screenshot:

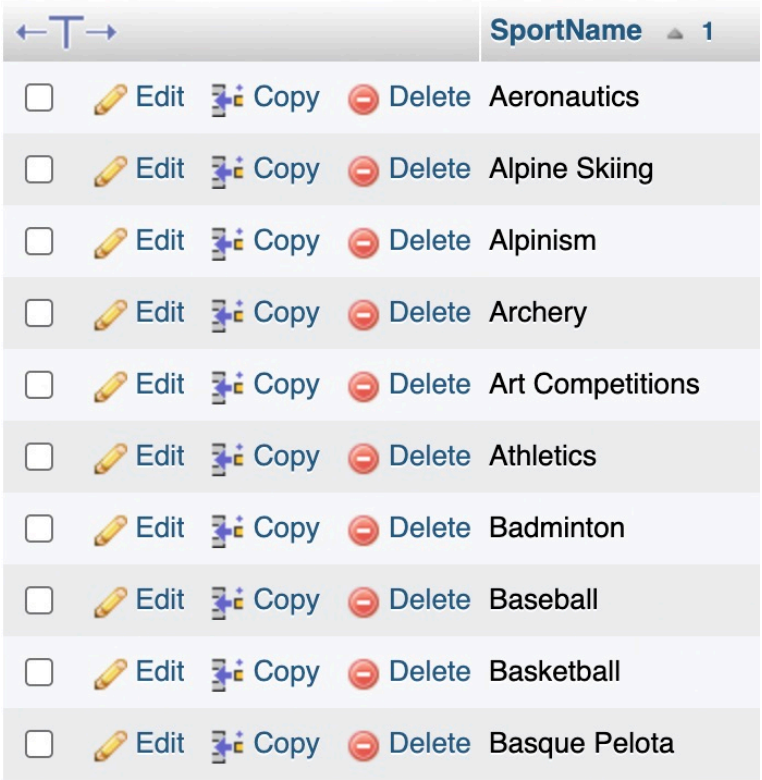
+ Options

←T→	AthleteID	AthleteName
Edit Copy Delete	1	Burch Devany
Edit Copy Delete	2	Dorothea Cescon
Edit Copy Delete	3	Barclay Benet
Edit Copy Delete	4	Sarah Smith
Edit Copy Delete	5	Carmella MacDermott
Edit Copy Delete	6	Lucilia Grebbin
Edit Copy Delete	7	Chrissy Thickens
Edit Copy Delete	8	Nadean Isaksson
Edit Copy Delete	9	Decca Markovich
Edit Copy Delete	10	Honoria Culp

Section A – SQL DML (SELECT)

Question 1	
Task	Return the names of all sports played at the Olympics (duplicates should not be included), ordered by SportName in alphabetical order.
Filename	*a1.sql* or *a1.txt*
SQL Solution	SELECT DISTINCT SportName FROM Sports ORDER BY SportName ASC;

Output screenshot:



←T→				SportName ▲ 1
☐	Edit	Copy	Delete	Aeronautics
☐	Edit	Copy	Delete	Alpine Skiing
☐	Edit	Copy	Delete	Alpinism
☐	Edit	Copy	Delete	Archery
☐	Edit	Copy	Delete	Art Competitions
☐	Edit	Copy	Delete	Athletics
☐	Edit	Copy	Delete	Badminton
☐	Edit	Copy	Delete	Baseball
☐	Edit	Copy	Delete	Basketball
☐	Edit	Copy	Delete	Basque Pelota

Question 2	
Task	Return the number of events that occurred for each sport during the month of July 2023.
Explanation	This query should return two columns, one for the SportID and one for the number of events that occurred for each sport.
Filename	*a2.sql* or *a2.txt*
SQL Solution	SELECT SportID, COUNT(EventID) AS NumberOfEvents FROM Events WHERE MONTH(Date) = 7 AND YEAR(Date) = 2023 GROUP BY SportID;

Output screenshot:

SportID	NumberOfEvents
27	2
10	1
49	1
2	1
31	1
4	1
18	1
58	1

Question 3	
Task	Return the number of medals won for each country.
Explanation	This query should return two columns, one for the CountryID, and one for the number of medals won (if the country has won 0 medals, it should still be included).
Filename	*a3.sql* or *a3.txt*
SQL Solution	SELECT C.CountryID, COUNT(M.AthleteID) AS NumberOfMedals FROM Countries C LEFT JOIN Athletes A ON C.CountryID = A.CountryID LEFT JOIN Medals M ON A.AthleteID = M.AthleteID GROUP BY C.CountryID;

Output screenshot:

CountryID	NumberOfMedals
1	0
2	107
3	0
4	1
5	0
6	2
7	0
8	0
9	1
10	0

Question 4	
Task	Return the medal tally for ‘Australia’ across all events.
Explanation	This query should return a table with two columns, one with the type of medal (Gold, Silver, or Bronze) and the other with the number of medals won for Australia.
Filename	*a4.sql* or *a4.txt*

SQL Solution	SELECT MedalType, COUNT(*) AS NumberOfMedals
	FROM Medals
	WHERE AthleteID IN (
	  SELECT AthleteID
	  FROM Athletes
	  WHERE CountryID = (
	  SELECT CountryID
	  FROM Countries
	  WHERE CountryName = 'Australia'))
	GROUP BY MedalType;

Output screenshot:

MedalType	NumberOfMedals
Gold	36
Bronze	38
Silver	33

Question 5	
Task	Return the country names of countries who have at least one participating athlete over the age of 30.
Explanation	This query should use at least one sub-query.
Filename	*a5.sql* or *a5.txt*
SQL Solution	SELECT CountryName FROM Countries WHERE CountryID IN (SELECT DISTINCT A.CountryID FROM Athletes A WHERE A.Age > 30);

Output screenshot:

←T→			CountryName
Edit	Copy	Delete	South Africa
Edit	Copy	Delete	Paraguay
Edit	Copy	Delete	Australia
Edit	Copy	Delete	Finland
Edit	Copy	Delete	Russia
Edit	Copy	Delete	United Kingdom
Edit	Copy	Delete	Tanzania
Edit	Copy	Delete	Lesotho
Edit	Copy	Delete	Kazakhstan
Edit	Copy	Delete	Germany

Question 6	
Task	Return the name, age of the youngest Australian athlete(s) participating in the Olympics.
Explanation	
Filename	*a6.sql* or *a6.txt*
SQL Solution	

```
SELECT AthleteName, Age
FROM Athletes
WHERE Age = (
    SELECT MIN(Age)
    FROM Athletes
    WHERE CountryID = (
        SELECT CountryID
        FROM Countries
        WHERE CountryName = 'Australia'));
```

Output screenshot:

				AthleteName	Age
☐	Edit	Copy	Delete	Conny Brettor	46
☐	Edit	Copy	Delete	Derrek Kitteringham	46
☐	Edit	Copy	Delete	Webster Barti	46
☐	Edit	Copy	Delete	Atlante Widdocks	46
☐	Edit	Copy	Delete	Gerome Creber	46

Question 7	
Task	Return the country names of countries that won more than one gold medal in the Olympics.
Explanation	
Filename	*a7.sql* or *a7.txt*
SQL Solution	SELECT C.CountryName FROM Countries C JOIN Athletes A ON C.CountryID = A.CountryID JOIN Medals M ON A.AthleteID = M.AthleteID WHERE M.MedalType = 'Gold' GROUP BY C.CountryName HAVING COUNT(*) > 1;

Output screenshot:

CountryName
Australia
Finland
United Kingdom
Lesotho
Egypt
Iran
Ethiopia
Jordan
Sierra Leone

Question 8	
Task	Return the names of athletes that medalled in expensive sports (i.e., sports that had at least 3 events with a ticket price over $100).
Explanation	**Hint**. You may want to use one or more views in your answer.
Filename	*a8.sql* or *a8.txt*
SQL Solution	CREATE VIEW ExpensiveSportCounts AS SELECT S.SportsID, COUNT(*) AS ExpensiveEventCount FROM Sports S JOIN Events E ON S.SportsID = E.SportID WHERE E.TicketPrice > 100 GROUP BY S.SportsID; SELECT DISTINCT A.AthleteName FROM Athletes A JOIN Medals M ON A.AthleteID = M.AthleteID JOIN Events E ON M.EventID = E.EventID JOIN ExpensiveSportCounts ES ON E.SportID = ES.SportsID WHERE ES.ExpensiveEventCount >= 3;

Output screenshot:

AthleteName
Carmella MacDermott
Derrek Kitteringham
Reagen Agglio
Blondie Quipp
Mora Meadley
Austen Lathaye
Minny Benadette
Carilyn Vacher
Issy Eyrl
Corabelle Bunker

Section B – SQL DML (UPDATE, DELETE, INSERT)

Question 1	
Task	Sarah Smith has had a positive performance-enhancing drugs test, so her medals (if any) need to be removed from the database.
Explanation	
Filename	*b1.sql* or *b1.txt*
SQL Solution	DELETE FROM Medals WHERE AthleteID = (SELECT AthleteID FROM Athletes WHERE AthleteName = 'Sarah Smith');

Output screenshot:

Show query box

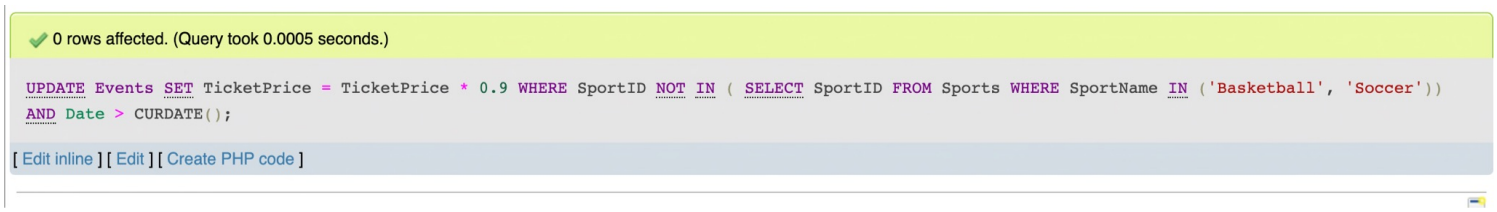
2 rows affected. (Query took 0.0003 seconds.)

```
DELETE FROM Medals WHERE AthleteID = ( SELECT AthleteID FROM Athletes WHERE AthleteName = 'Sarah Smith');
```

[Edit inline] [Edit] [Create PHP code]

Question 2	
Task	The ticket price for all sports and games except Basketball and Soccer are to be reduced by 10% due to a lack of demand. Issue this update in the database.
Explanation	This query should update the price of all other events in the future (i.e., the Date is later than the current date), to be 10% less than the existing price in the database.
Filename	*b2.sql* or *b2.txt*
SQL Solution	UPDATE Events SET TicketPrice = TicketPrice * 0.9 WHERE SportID NOT IN (SELECT SportID FROM Sports WHERE SportName IN ('Basketball', 'Soccer')) AND Date > CURDATE();

Output screenshot:

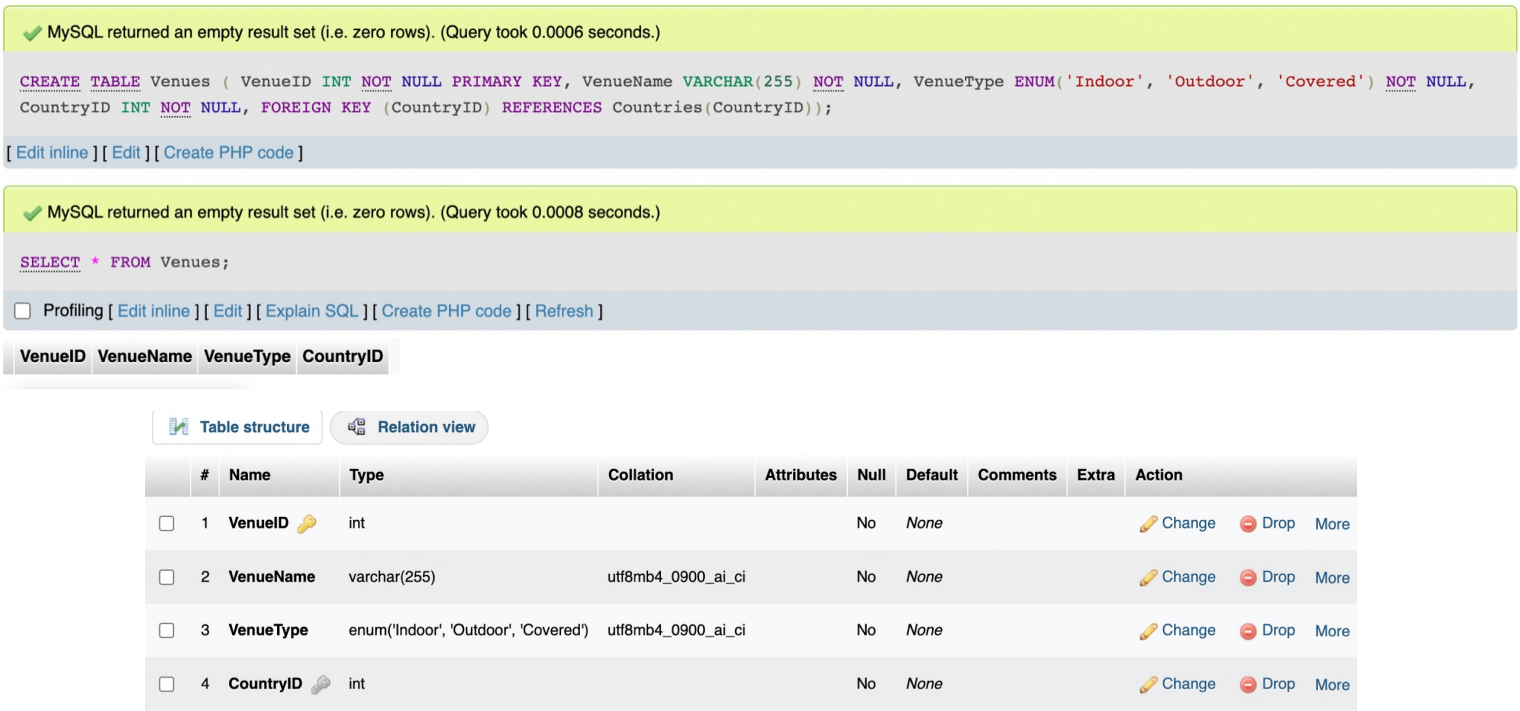


Section C – SQL DDL

<table>
<tr><th colspan="2">Question 1</th></tr>
<tr><td>Task</td><td>Write a SQL DDL query to implement the following relational schema and associated foreign keys.</td></tr>
<tr><td>Explanation</td><td>The relational schema for this the table is as follows:
<table>
<tr><th colspan="4">Table: Venues</th></tr>
<tr><th>Column</th><th>Data Type</th><th>Allow Nulls?</th><th>Primary Key?</th></tr>
<tr><td>VenueID</td><td>INT</td><td>N</td><td>Y</td></tr>
<tr><td>VenueName</td><td>VARCHAR</td><td>N</td><td>N</td></tr>
<tr><td>VenueType</td><td>(‘Indoor’, ‘Outdoor’, ‘Covered’)</td><td>N</td><td>N</td></tr>
<tr><td>CountryID</td><td>INT</td><td>N</td><td>N</td></tr>
</table></td></tr>
<tr><td>Filename</td><td>c1.sql or c1.txt</td></tr>
<tr><td>SQL Solution</td><td>
<pre>CREATE TABLE Venues (
    VenueID INT NOT NULL PRIMARY KEY,
    VenueName VARCHAR(255) NOT NULL,
    VenueType ENUM('Indoor', 'Outdoor', 'Covered') NOT NULL,
    CountryID INT NOT NULL,
    FOREIGN KEY (CountryID) REFERENCES Countries(CountryID));

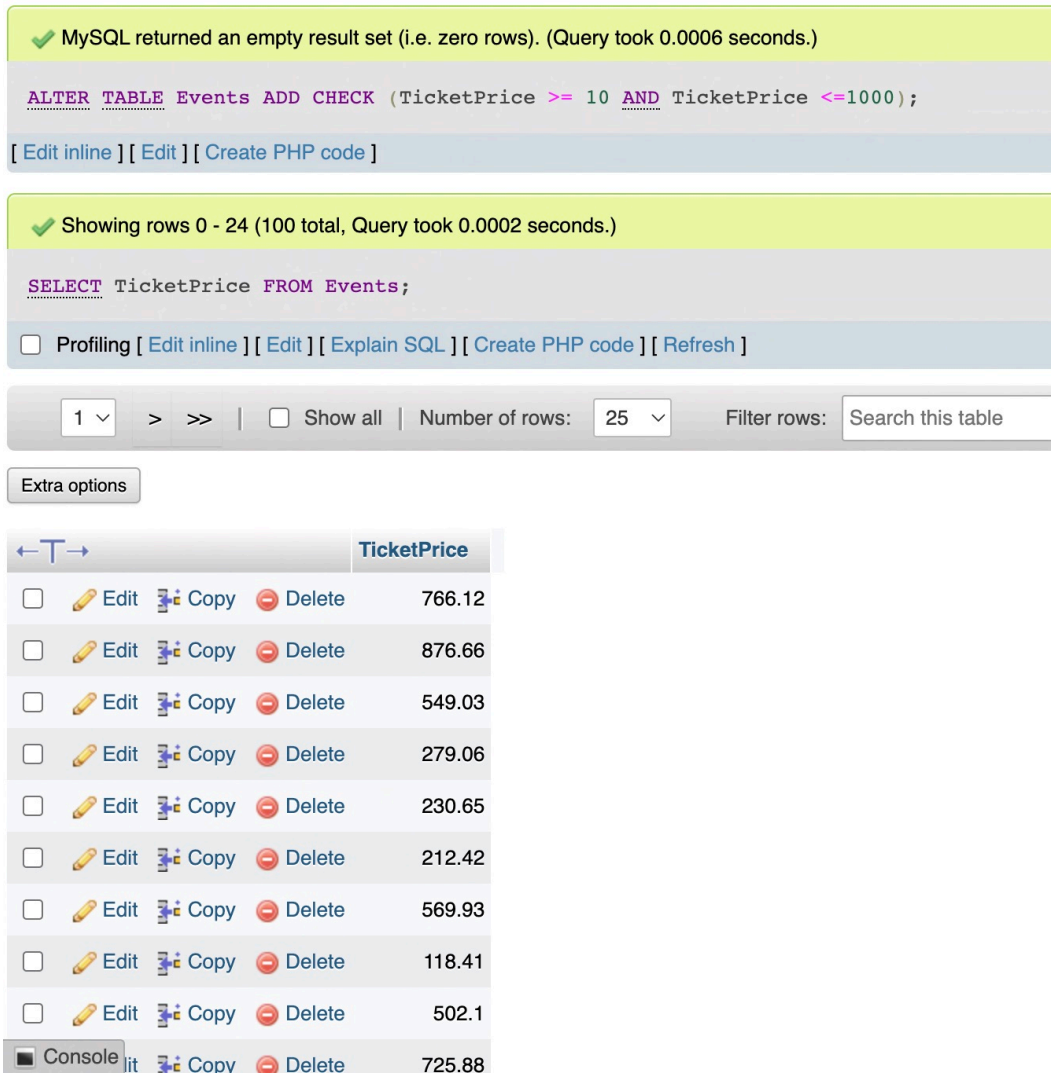
SELECT *
FROM Venues;</pre>
</td></tr>
</table>

Output screenshot:



Question 2	
Task	To ensure that all events are reasonably priced, add a constraint that ensures that no ticket is priced under $10 and over $1000.
Explanation	The following resources may be useful when answering this question: Check constraints
Filename	*c2.sql* or c2.*txt*
SQL Solution	ALTER TABLE Events ADD CHECK (TicketPrice >= 10 AND TicketPrice <=1000); SELECT TicketPrice FROM Events;

Output screenshot:



Section D – Critical Thinking

In this section, you will receive theoretical situations related to the UoD mentioned in the task description. Your task is to offer strategies to tackle the situation and write SQL queries to execute the approaches.

- INFS1200 students answer Question 1 only.
- INFS7900 students answer both Question 1 and Question 2.

Question 1	
Task	Olympics games planners want to know what to expect during the any of the most busy weeks in the Olympics (i.e., how many athletes will be participating, how many different sports, how many different countries and so on). Propose a strategy for the given task and write an SQL query to implement that strategy. **Hint**: The SQL WEEK() function may be useful.
Strategies	First, we need to alter the table 'Athletes' to include EventID. This allows us to know which events each athlete is participating in, as well as the countries that participate in those events through Athletes(CountryID). Hence we are able to calculate the count of athletes, events, and countries participating in each week. We then order the busiest week as the one with the most athletes. In the case of the same amount of athletes, the earliest week is placed first.

SQL Solution	ALTER TABLE Athletes
	ADD EventID INT;
	
	SELECT
	  WEEK(Events.Date) AS OlympicWeek,
	  COUNT(DISTINCT Athletes.AthleteID) AS NumberOfAthletes,
	  COUNT(DISTINCT Events.SportID) AS NumberOfSports,
	  COUNT(DISTINCT Athletes.CountryID) AS NumberOfCountries
	FROM
	  Events
	INNER JOIN
	  Athletes ON Events.EventID = Athletes.EventID
	GROUP BY
	  OlympicWeek
	ORDER BY
	  NumberOfAthletes DESC, OlympicWeek ASC;