Master of Information Technology (MIT)

Data Management Systems

Assignment:	Database Design and Modelling
Due Date:	**********
Submission:	Submit your report including an ERD design in a GROUP via MySCU
Weight:	25% of overall unit assessment

Entity List:

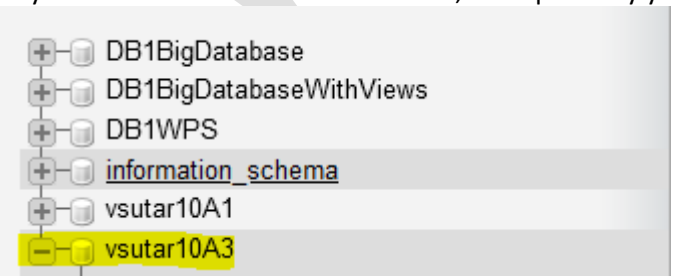
- Coach
- Country
- Event
- Event Registration
- Medical Officer
- People
- Pools
- Race
- Race Swimmer
- Swimmer
- Team Registration

People is the super type and its sub type includes Swimmer, Medical Officer and Coach.

B. Assessment Requirements

B.1 Database Development and Relationships **4 MARKS**

Build your database in MySQL using PhpMyAdmin. Your database should be created on the Infotech server and be named with your username followed by A3 (e.g., aalaeiA3). Referential integrity and any other constraints must be created, as required by your design.




```
CREATE TABLE `coach` (
  `PersonID` int(11) NOT NULL,
  `CertificationLevel` int(11) NOT NULL,
  `CertificationDate` date NOT NULL,
  `WWCDate` date NOT NULL
```

Writing Sol

```
);

CREATE TABLE `country` (
  `CountryCode` char(4) NOT NULL,
  `Country` varchar(250) NOT NULL,
  `MainContact` varchar(50) DEFAULT NULL
) ;
CREATE TABLE `event` (
  `EventID` int(11) NOT NULL,
  `EventName` varchar(500) NOT NULL,
  `MinimumQualifyingTime` varchar(20) NOT NULL
) ;
CREATE TABLE `event_registeration` (
  `EventID` int(11) NOT NULL,
  `SwimmerID` int(11) NOT NULL,
  `QualifyingTime` varchar(25) NOT NULL,
  `QualifyingDate` date NOT NULL,
  `QualifyingCompetition` text NOT NULL
);
CREATE TABLE `medical_officer` (
  `PersonID` int(11) NOT NULL,
  `Qualification` varchar(250) NOT NULL,
  `QualificationYear` int(11) NOT NULL,
  `Specialisation` varchar(450) DEFAULT NULL
) ;
CREATE TABLE `people` (
  `PersonID` int(11) NOT NULL,
  `FirstName` varchar(25) NOT NULL,
  `LastName` varchar(25) NOT NULL,
  `CountryCode` char(4) NOT NULL,
  `Address` text NOT NULL,
  `City` varchar(150) NOT NULL,
  `State` char(4) NOT NULL,
  `PostCode` int(11) NOT NULL,
  `Mobile` varchar(25) NOT NULL,
  `ContactType` varchar(40) NOT NULL
) ;
CREATE TABLE `pools` (
  `PoolNumber` int(11) NOT NULL,
  `PoolName` varchar(100) NOT NULL,
  `LaneCount` int(11) DEFAULT NULL,
  `Length` int(11) NOT NULL
);
CREATE TABLE `race` (
  `RaceID` int(11) NOT NULL,
  `EventID` int(11) NOT NULL,
  `RaceName` varchar(250) NOT NULL,
  `Day` int(11) NOT NULL,
  `StartTime` varchar(25) NOT NULL,
  `Location` int(11) NOT NULL
) ;

CREATE TABLE `race_swimmer` (
```

Writing Sol

```
  `RaceID` int(11) NOT NULL,
  `SwimmerID` int(11) NOT NULL,
  `LaneNumber` int(11) NOT NULL,
  `RecordedTime` varchar(25) NOT NULL,
  `Place` int(11) DEFAULT NULL
) ;

CREATE TABLE `swimmer` (
  `PersonID` int(11) NOT NULL,
  `Gender` char(7) NOT NULL,
  `DOB` date NOT NULL,
  `CoachID` int(11) NOT NULL,
  `TeamID` int(11) NOT NULL
) ;
CREATE TABLE `team_registeration` (
  `TeamID` int(11) NOT NULL,
  `CountryCode` char(4) NOT NULL,
  `CoachID` int(11) NOT NULL,
  `MedicalOfficerID` int(11) NOT NULL,
  `TeamManagerID` int(11) NOT NULL,
  `TeamLeaderID` int(11) NOT NULL
) ;
```

B.2 Data Type and Naming **4 MARKS**

Create a MySQL database using phpMyAdmin. **You must name this database as your username followed by A3.** For example: falaei10A3.
Choice and consistency of table and attribute naming conventions used, selection of data types, primary and foreign keys.

```
ALTER TABLE `coach`
  ADD PRIMARY KEY (`PersonID`);

--
-- Indexes for table `country`
--
ALTER TABLE `country`
  ADD PRIMARY KEY (`CountryCode`);

--
-- Indexes for table `event`
--
ALTER TABLE `event`
  ADD PRIMARY KEY (`EventID`);

--
-- Indexes for table `event_registeration`
--
ALTER TABLE `event_registeration`
  ADD PRIMARY KEY (`EventID`,`SwimmerID`),
  ADD KEY `FK_EVENT_SWIMMER` (`SwimmerID`);

--
-- Indexes for table `medical_officer`
--
```

Writing Sol

```
ALTER TABLE `medical_officer`
  ADD PRIMARY KEY (`PersonID`);

--
-- Indexes for table `people`
--
ALTER TABLE `people`
  ADD PRIMARY KEY (`PersonID`),
  ADD KEY `FK_COUNTRY_PEOPLE` (`CountryCode`);

--
-- Indexes for table `pools`
--
ALTER TABLE `pools`
  ADD PRIMARY KEY (`PoolNumber`);

--
-- Indexes for table `race`
--
ALTER TABLE `race`
  ADD PRIMARY KEY (`RaceID`),
  ADD KEY `FK_RACE_EVENT` (`EventID`),
  ADD KEY `FK_RACE_LOCATION` (`Location`);

--
-- Indexes for table `race_swimmer`
--
ALTER TABLE `race_swimmer`
  ADD PRIMARY KEY (`RaceID`,`SwimmerID`),
  ADD KEY `FK_RACE_SWIMMER` (`SwimmerID`);

--
-- Indexes for table `swimmer`
--
ALTER TABLE `swimmer`
  ADD PRIMARY KEY (`PersonID`),
  ADD KEY `FK_TEAM_SWIMMER` (`TeamID`),
  ADD KEY `FK_SWIMMER_COACH` (`CoachID`);

--
-- Indexes for table `team_registeration`
--
ALTER TABLE `team_registeration`
  ADD PRIMARY KEY (`TeamID`),
  ADD KEY `FK_COUNTRY_TEAM` (`CountryCode`),
  ADD KEY `FK_COACH_TEAM` (`CoachID`),
  ADD KEY `FK_MEDICAL_OFFICER_TEAM` (`MedicalOfficerID`),
  ADD KEY `FK_TEAM_MANAGER_TEAM` (`TeamLeaderID`),
  ADD KEY `FK_TEAM_LEADER_TEAM` (`TeamManagerID`);

--
-- AUTO_INCREMENT for dumped tables
--
```

Writing Sol

```
--
-- AUTO_INCREMENT for table `event`
--
ALTER TABLE `event`
  MODIFY `EventID` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=41;

--
-- AUTO_INCREMENT for table `people`
--
ALTER TABLE `people`
  MODIFY `PersonID` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=74;

--
-- AUTO_INCREMENT for table `pools`
--
ALTER TABLE `pools`
  MODIFY `PoolNumber` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=5;

--
-- AUTO_INCREMENT for table `race`
--
ALTER TABLE `race`
  MODIFY `RaceID` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=15;

--
-- AUTO_INCREMENT for table `team_registeration`
--
ALTER TABLE `team_registeration`
  MODIFY `TeamID` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=11;

--
-- Constraints for dumped tables
--

--
-- Constraints for table `coach`
--
ALTER TABLE `coach`
  ADD CONSTRAINT `FK_COACH_PEOPLE` FOREIGN KEY (`PersonID`) REFERENCES `people` (`PersonID`);

--
-- Constraints for table `event_registeration`
--
ALTER TABLE `event_registeration`
  ADD CONSTRAINT `FK_EVENT_SWIMMER` FOREIGN KEY (`SwimmerID`) REFERENCES `swimmer` (`PersonID`),
  ADD CONSTRAINT `FK_EVEN_REGISTERATION` FOREIGN KEY (`EventID`) REFERENCES `event` (`EventID`);

--
-- Constraints for table `medical_officer`
```

Writing Sol

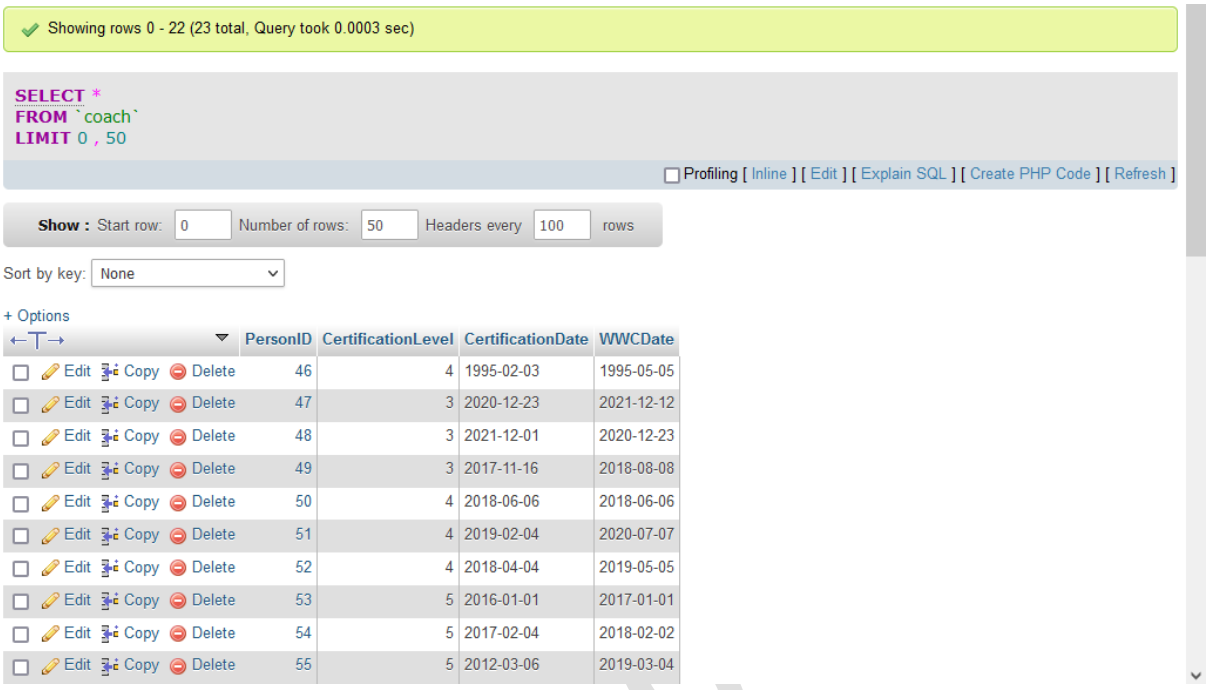
```
--
ALTER TABLE `medical_officer`
 ADD CONSTRAINT `FK_MEDICAL_OFFICER_PERSON` FOREIGN KEY (`PersonID`) REFERENCES `people` (`PersonID`);

--
-- Constraints for table `people`
--
ALTER TABLE `people`
 ADD CONSTRAINT `FK_COUNTRY_PEOPLE` FOREIGN KEY (`CountryCode`) REFERENCES `country` (`CountryCode`);

--
-- Constraints for table `race`
--
ALTER TABLE `race`
 ADD CONSTRAINT `FK_RACE_EVENT` FOREIGN KEY (`EventID`) REFERENCES `event` (`EventID`),
 ADD CONSTRAINT `FK_RACE_LOCATION` FOREIGN KEY (`Location`) REFERENCES `pools` (`PoolNumber`);

--
-- Constraints for table `race_swimmer`
--
ALTER TABLE `race_swimmer`
 ADD CONSTRAINT `FK_RACE_RACE_SWIMMER` FOREIGN KEY (`RaceID`) REFERENCES `race` (`RaceID`),
 ADD CONSTRAINT `FK_RACE_SWIMMER` FOREIGN KEY (`SwimmerID`) REFERENCES `swimmer` (`PersonID`);

--
-- Constraints for table `swimmer`
--
ALTER TABLE `swimmer`
 ADD CONSTRAINT `FK_SWIMMER_COACH` FOREIGN KEY (`CoachID`) REFERENCES `people` (`PersonID`),
 ADD CONSTRAINT `FK_SWIMMER_PEOPLE` FOREIGN KEY (`PersonID`) REFERENCES `people` (`PersonID`),
 ADD CONSTRAINT `FK_TEAM_SWIMMER` FOREIGN KEY (`TeamID`) REFERENCES `team_registeration` (`TeamID`);

--
-- Constraints for table `team_registeration`
--
ALTER TABLE `team_registeration`
 ADD CONSTRAINT `FK_COACH_TEAM` FOREIGN KEY (`CoachID`) REFERENCES `people` (`PersonID`),
 ADD CONSTRAINT `FK_COUNTRY_TEAM` FOREIGN KEY (`CountryCode`) REFERENCES `country` (`CountryCode`),
 ADD CONSTRAINT `FK_MEDICAL_OFFICER_TEAM` FOREIGN KEY (`MedicalOfficerID`) REFERENCES `people` (`PersonID`),
 ADD CONSTRAINT `FK_TEAM_LEADER_TEAM` FOREIGN KEY (`TeamManagerID`) REFERENCES `people` (`PersonID`),
 ADD CONSTRAINT `FK_TEAM_MANAGER_TEAM` FOREIGN KEY (`TeamLeaderID`) REFERENCES `people` (`PersonID`);
```

Writing Sol

COMMIT;

B.3 Test/Sample Data 2.5 MARKS

You must provide enough valid data in your database to run the SQL queries below successfully with at least 5-10 rows in the result set.



Showing rows 0 - 22 (23 total, Query took 0.0003 sec)

```
SELECT *
FROM `coach`
LIMIT 0 , 50
```

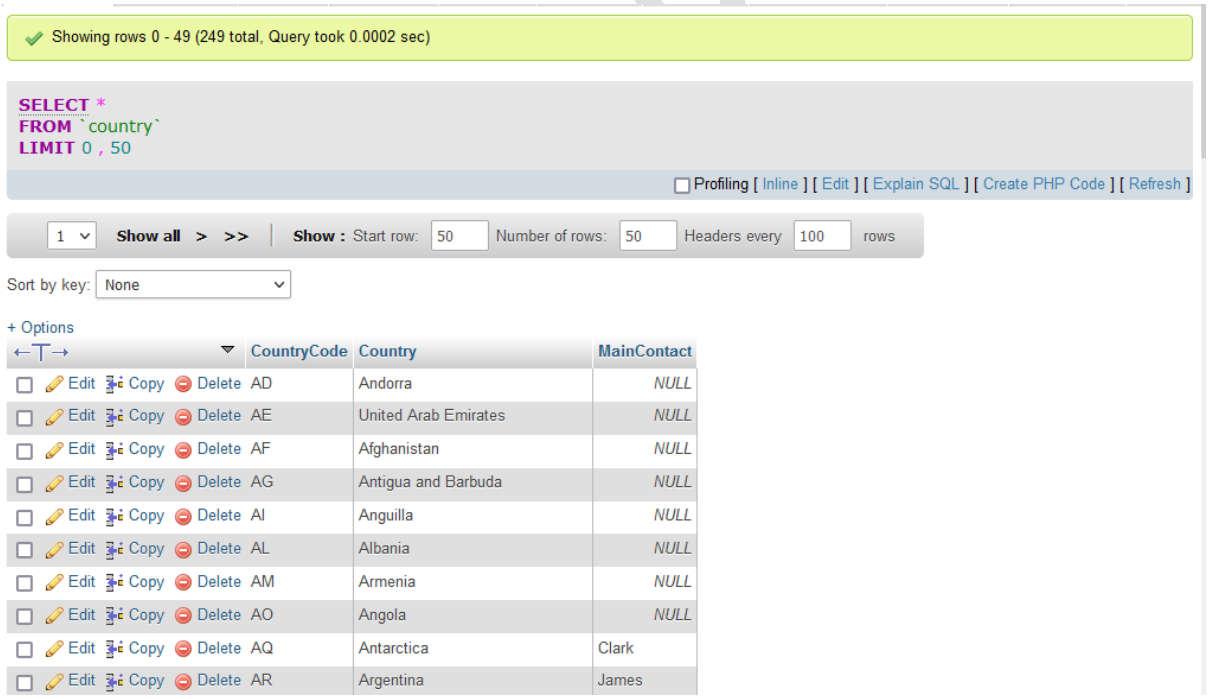
Profiling [Inline] [Edit] [Explain SQL] [Create PHP Code] [Refresh]

Show : Start row: 0 Number of rows: 50 Headers every 100 rows

Sort by key: None

+ Options

	PersonID	CertificationLevel	CertificationDate	WWCDate
Edit Copy Delete	46	4	1995-02-03	1995-05-05
Edit Copy Delete	47	3	2020-12-23	2021-12-12
Edit Copy Delete	48	3	2021-12-01	2020-12-23
Edit Copy Delete	49	3	2017-11-16	2018-08-08
Edit Copy Delete	50	4	2018-06-06	2018-06-06
Edit Copy Delete	51	4	2019-02-04	2020-07-07
Edit Copy Delete	52	4	2018-04-04	2019-05-05
Edit Copy Delete	53	5	2016-01-01	2017-01-01
Edit Copy Delete	54	5	2017-02-04	2018-02-02
Edit Copy Delete	55	5	2012-03-06	2019-03-04



Showing rows 0 - 49 (249 total, Query took 0.0002 sec)

```
SELECT *
FROM `country`
LIMIT 0 , 50
```

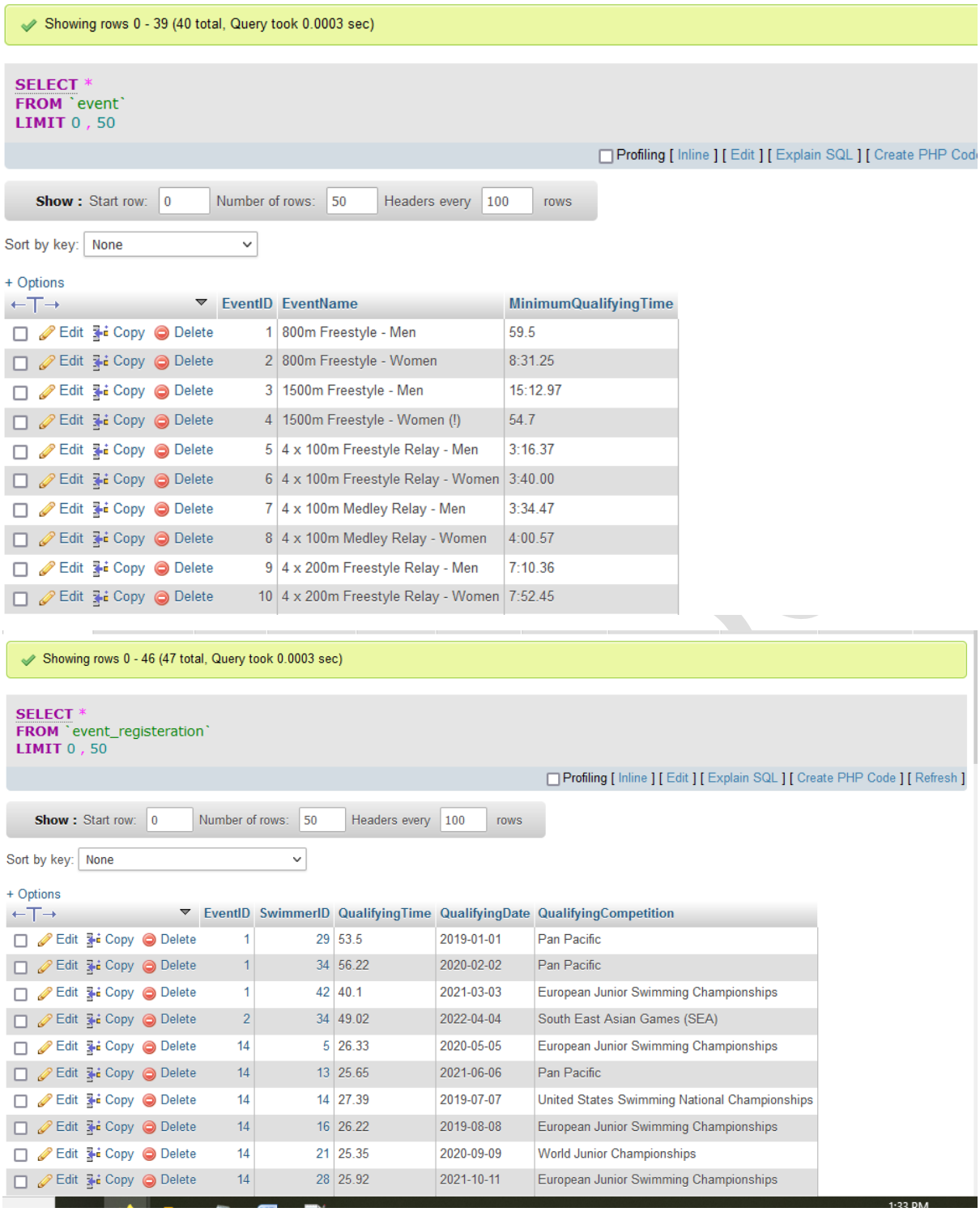
Profiling [Inline] [Edit] [Explain SQL] [Create PHP Code] [Refresh]

1 Show all > >> | Show : Start row: 50 Number of rows: 50 Headers every 100 rows

Sort by key: None

+ Options

	CountryCode	Country	MainContact
Edit Copy Delete	AD	Andorra	NULL
Edit Copy Delete	AE	United Arab Emirates	NULL
Edit Copy Delete	AF	Afghanistan	NULL
Edit Copy Delete	AG	Antigua and Barbuda	NULL
Edit Copy Delete	AI	Anguilla	NULL
Edit Copy Delete	AL	Albania	NULL
Edit Copy Delete	AM	Armenia	NULL
Edit Copy Delete	AO	Angola	NULL
Edit Copy Delete	AQ	Antarctica	Clark
Edit Copy Delete	AR	Argentina	James



Showing rows 0 - 39 (40 total, Query took 0.0003 sec)

```
SELECT *
FROM `event`
LIMIT 0 , 50
```

Profiling [Inline] [Edit] [Explain SQL] [Create PHP Cod

Show : Start row: 0 Number of rows: 50 Headers every 100 rows

Sort by key: None

+ Options

	EventID	EventName	MinimumQualifyingTime
Edit Copy Delete	1	800m Freestyle - Men	59.5
Edit Copy Delete	2	800m Freestyle - Women	8:31.25
Edit Copy Delete	3	1500m Freestyle - Men	15:12.97
Edit Copy Delete	4	1500m Freestyle - Women (!)	54.7
Edit Copy Delete	5	4 x 100m Freestyle Relay - Men	3:16.37
Edit Copy Delete	6	4 x 100m Freestyle Relay - Women	3:40.00
Edit Copy Delete	7	4 x 100m Medley Relay - Men	3:34.47
Edit Copy Delete	8	4 x 100m Medley Relay - Women	4:00.57
Edit Copy Delete	9	4 x 200m Freestyle Relay - Men	7:10.36
Edit Copy Delete	10	4 x 200m Freestyle Relay - Women	7:52.45

Showing rows 0 - 46 (47 total, Query took 0.0003 sec)

```
SELECT *
FROM `event_registeration`
LIMIT 0 , 50
```

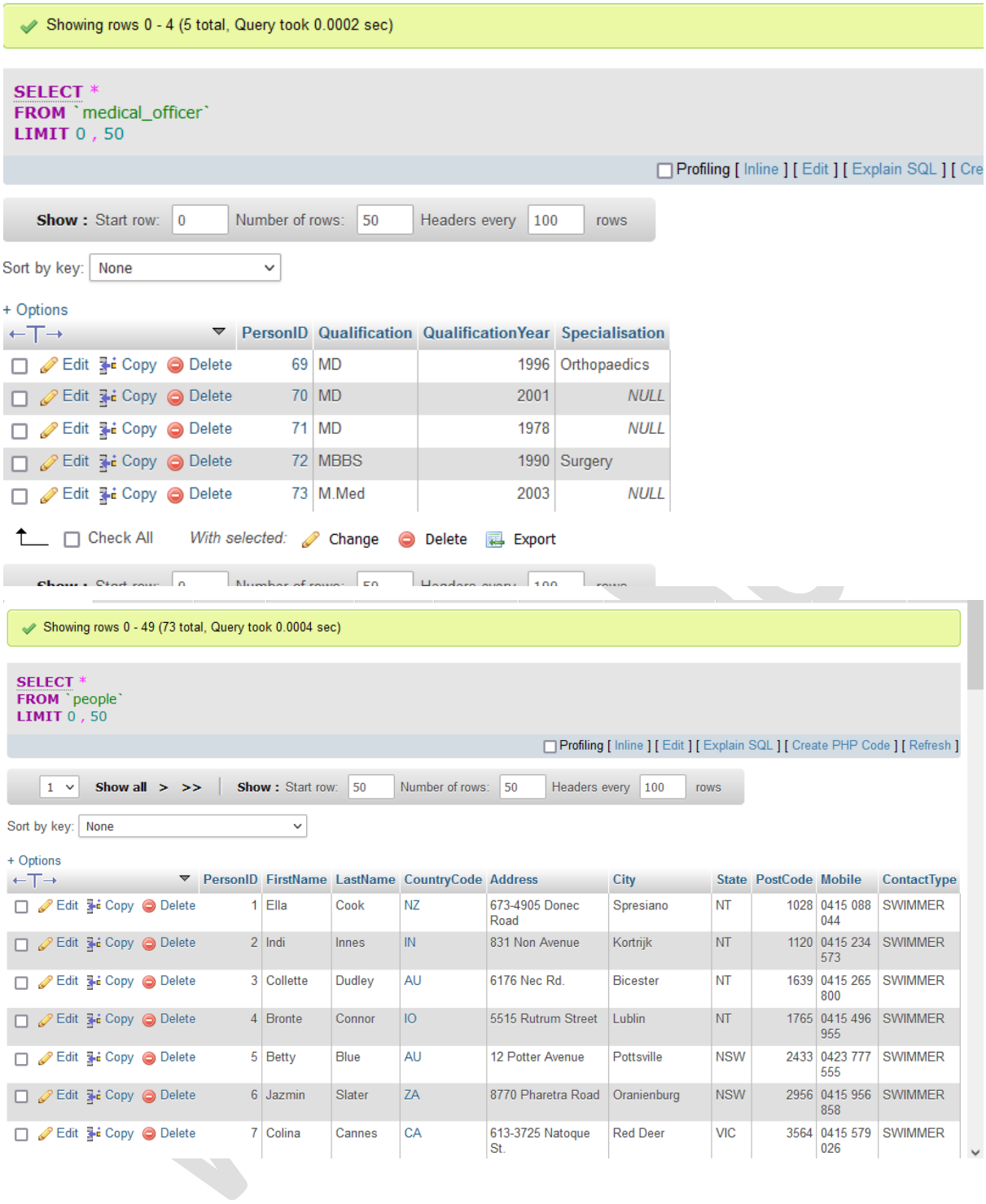
Profiling [Inline] [Edit] [Explain SQL] [Create PHP Code] [Refresh]

Show : Start row: 0 Number of rows: 50 Headers every 100 rows

Sort by key: None

+ Options

	EventID	SwimmerID	QualifyingTime	QualifyingDate	QualifyingCompetition
Edit Copy Delete	1	29	53.5	2019-01-01	Pan Pacific
Edit Copy Delete	1	34	56.22	2020-02-02	Pan Pacific
Edit Copy Delete	1	42	40.1	2021-03-03	European Junior Swimming Championships
Edit Copy Delete	2	34	49.02	2022-04-04	South East Asian Games (SEA)
Edit Copy Delete	14	5	26.33	2020-05-05	European Junior Swimming Championships
Edit Copy Delete	14	13	25.65	2021-06-06	Pan Pacific
Edit Copy Delete	14	14	27.39	2019-07-07	United States Swimming National Championships
Edit Copy Delete	14	16	26.22	2019-08-08	European Junior Swimming Championships
Edit Copy Delete	14	21	25.35	2020-09-09	World Junior Championships
Edit Copy Delete	14	28	25.92	2021-10-11	European Junior Swimming Championships



Showing rows 0 - 4 (5 total, Query took 0.0002 sec)

```
SELECT *
FROM `medical_officer`
LIMIT 0 , 50
```

Profiling [Inline] [Edit] [Explain SQL] [Cre

Show : Start row: 0 Number of rows: 50 Headers every 100 rows

Sort by key: None

+ Options

	PersonID	Qualification	QualificationYear	Specialisation
Edit Copy Delete	69	MD	1996	Orthopaedics
Edit Copy Delete	70	MD	2001	NULL
Edit Copy Delete	71	MD	1978	NULL
Edit Copy Delete	72	MBBS	1990	Surgery
Edit Copy Delete	73	M.Med	2003	NULL

Check All With selected: Change Delete Export

Showing rows 0 - 49 (73 total, Query took 0.0004 sec)

```
SELECT *
FROM `people`
LIMIT 0 , 50
```

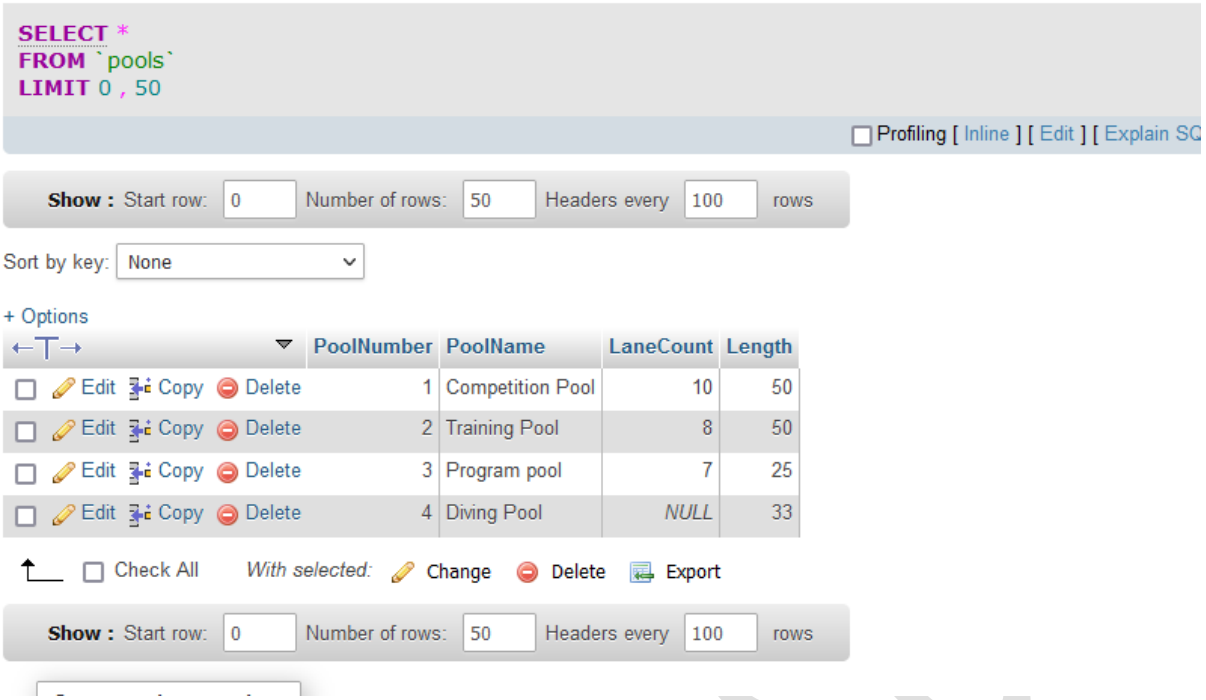
Profiling [Inline] [Edit] [Explain SQL] [Create PHP Code] [Refresh]

1 Show all > >> | Show : Start row: 50 Number of rows: 50 Headers every 100 rows

Sort by key: None

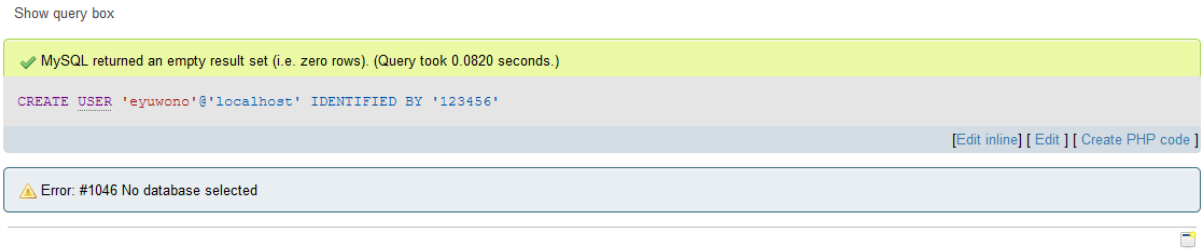
+ Options

	PersonID	FirstName	LastName	CountryCode	Address	City	State	PostCode	Mobile	ContactType
Edit Copy Delete	1	Ella	Cook	NZ	673-4905 Donec Road	Spresiano	NT	1028	0415 088 044	SWIMMER
Edit Copy Delete	2	Indi	Innes	IN	831 Non Avenue	Kortrijk	NT	1120	0415 234 573	SWIMMER
Edit Copy Delete	3	Collette	Dudley	AU	6176 Nec Rd.	Bicester	NT	1639	0415 265 800	SWIMMER
Edit Copy Delete	4	Bronte	Connor	IO	5515 Rutrum Street	Lublin	NT	1765	0415 496 955	SWIMMER
Edit Copy Delete	5	Betty	Blue	AU	12 Potter Avenue	Pottsville	NSW	2433	0423 777 555	SWIMMER
Edit Copy Delete	6	Jazmin	Slater	ZA	8770 Pharetra Road	Oranienburg	NSW	2956	0415 956 858	SWIMMER
Edit Copy Delete	7	Colina	Cannes	CA	613-3725 Natoque St.	Red Deer	VIC	3564	0415 579 026	SWIMMER

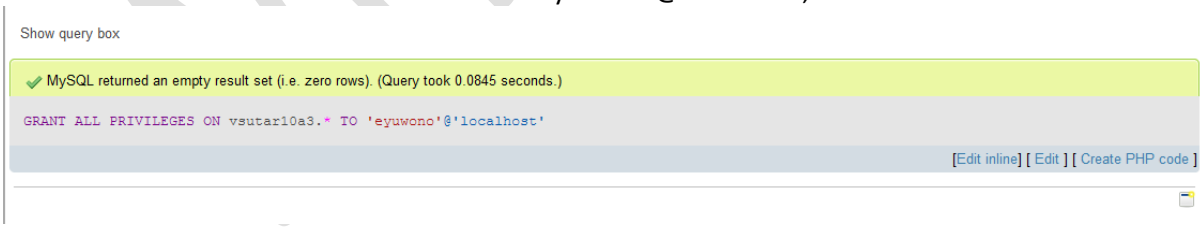


B.4 Privileges **2 MARKS**

Provide access to your database for the user eyuwono or another student user account of your choice. Details of how to do this can be seen in the provided screencast.
CREATE USER 'eyuwono'@'localhost' IDENTIFIED BY '123456';

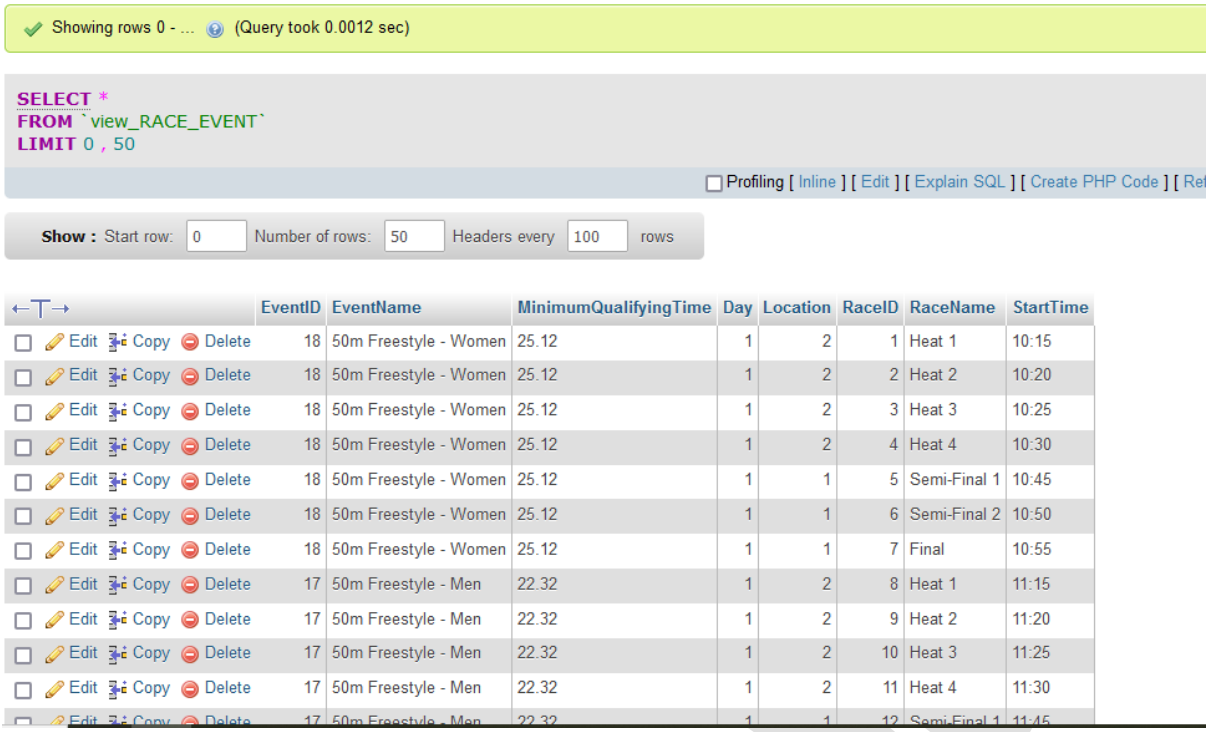


GRANT ALL PRIVILEGES ON vsutar10a3.* TO 'eyuwono'@'localhost';



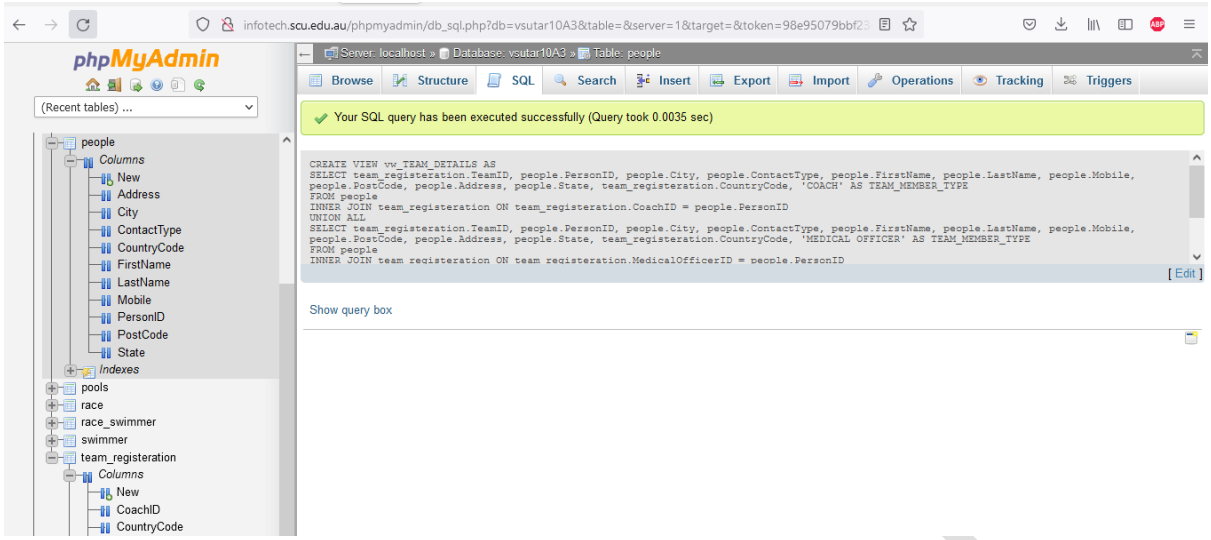
B.5 Use of Views **1.5 MARKS**

The database should be capable of producing the query results below. To get the required results, you may need to build underlying queries or views. Create at least four views and where appropriate, these views should incorporate the use of standard practices, informative column headings that fit the expected use of the view.
CREATE VIEW view_RACE_EVENT AS
SELECT event.EventID, event.EventName,
event.MinimumQualifyingTime,race.Day,race.Location,race.RaceID,race.RaceName,race.StartTime
FROM event INNER JOIN race ON event.EventID=race.EventID;

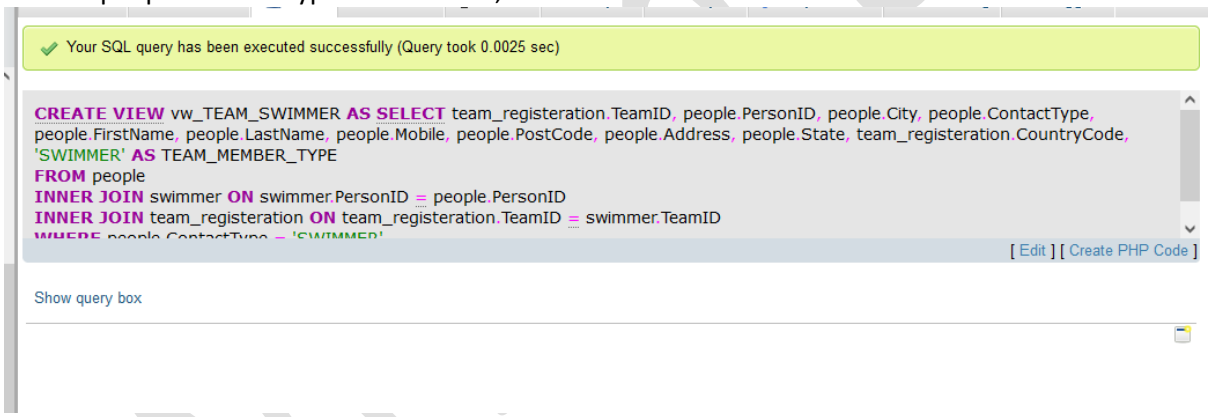


	EventID	EventName	MinimumQualifyingTime	Day	Location	RaceID	RaceName	StartTime
Edit Copy Delete	18	50m Freestyle - Women	25.12	1	2	1	Heat 1	10:15
Edit Copy Delete	18	50m Freestyle - Women	25.12	1	2	2	Heat 2	10:20
Edit Copy Delete	18	50m Freestyle - Women	25.12	1	2	3	Heat 3	10:25
Edit Copy Delete	18	50m Freestyle - Women	25.12	1	2	4	Heat 4	10:30
Edit Copy Delete	18	50m Freestyle - Women	25.12	1	1	5	Semi-Final 1	10:45
Edit Copy Delete	18	50m Freestyle - Women	25.12	1	1	6	Semi-Final 2	10:50
Edit Copy Delete	18	50m Freestyle - Women	25.12	1	1	7	Final	10:55
Edit Copy Delete	17	50m Freestyle - Men	22.32	1	2	8	Heat 1	11:15
Edit Copy Delete	17	50m Freestyle - Men	22.32	1	2	9	Heat 2	11:20
Edit Copy Delete	17	50m Freestyle - Men	22.32	1	2	10	Heat 3	11:25
Edit Copy Delete	17	50m Freestyle - Men	22.32	1	2	11	Heat 4	11:30
Edit Copy Delete	17	50m Freestyle - Men	22.32	1	1	12	Semi-Final 1	11:45

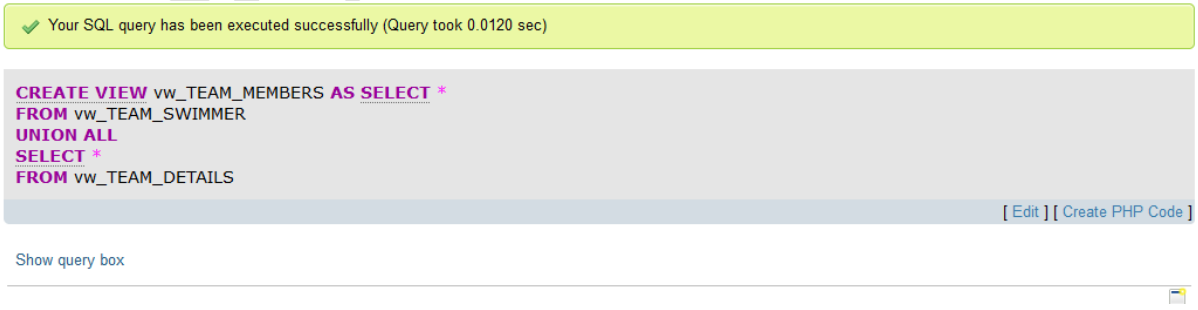
```
CREATE VIEW vw_TEAM_DETAILS AS
SELECT team_registeration.TeamID, people.PersonID, people.City, people.ContactType,
people.FirstName, people.LastName, people.Mobile, people.PostCode, people.Address,
people.State, team_registeration.CountryCode, 'COACH' AS TEAM_MEMBER_TYPE
FROM people
INNER JOIN team_registeration ON team_registeration.CoachID = people.PersonID
UNION ALL
SELECT team_registeration.TeamID, people.PersonID, people.City, people.ContactType,
people.FirstName, people.LastName, people.Mobile, people.PostCode, people.Address,
people.State, team_registeration.CountryCode, 'MEDICAL OFFICER' AS TEAM_MEMBER_TYPE
FROM people
INNER JOIN team_registeration ON team_registeration.MedicalOfficerID = people.PersonID
UNION ALL
SELECT team_registeration.TeamID, people.PersonID, people.City, people.ContactType,
people.FirstName, people.LastName, people.Mobile, people.PostCode, people.Address,
people.State, team_registeration.CountryCode, 'TEAM LEADER' AS TEAM_MEMBER_TYPE
FROM people
INNER JOIN team_registeration ON team_registeration.TeamLeaderID = people.PersonID
UNION ALL
SELECT team_registeration.TeamID, people.PersonID, people.City, people.ContactType,
people.FirstName, people.LastName, people.Mobile, people.PostCode, people.Address,
people.State, team_registeration.CountryCode, 'TEAM MANAGER' AS TEAM_MEMBER_TYPE
FROM people
INNER JOIN team_registeration ON team_registeration.TeamManagerID = people.PersonID;
```



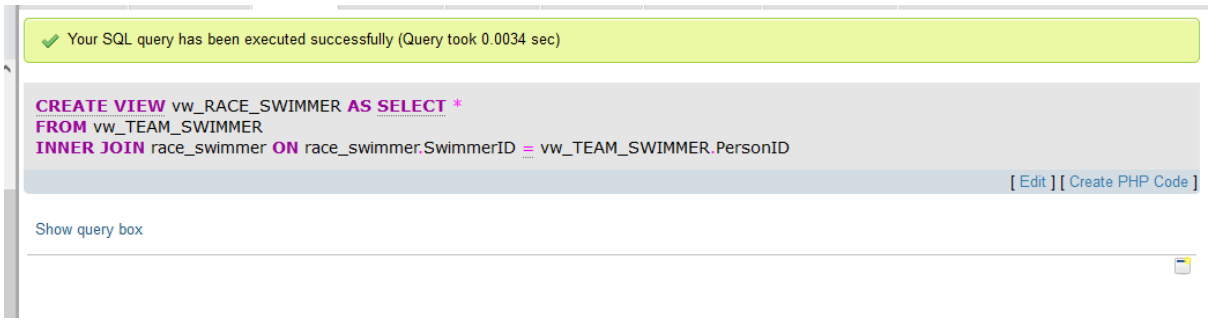
CREATE VIEW vw_TEAM_SWIMMER AS
SELECT team_registeration.TeamID,
people.PersonID, people.City, people.ContactType, people.FirstName, people.LastName,
people.Mobile, people.PostCode, people.Address, people.State,
team_registeration.CountryCode, 'SWIMMER' AS TEAM_MEMBER_TYPE
FROM people
INNER JOIN swimmer ON swimmer.PersonID=people.PersonID
INNER JOIN team_registeration ON team_registeration.TeamID=swimmer.TeamID
WHERE people.ContactType='SWIMMER';



CREATE VIEW vw_TEAM_MEMBERS AS
SELECT * FROM vw_TEAM_SWIMMER
UNION ALL
SELECT * FROM vw_TEAM_DETAILS;

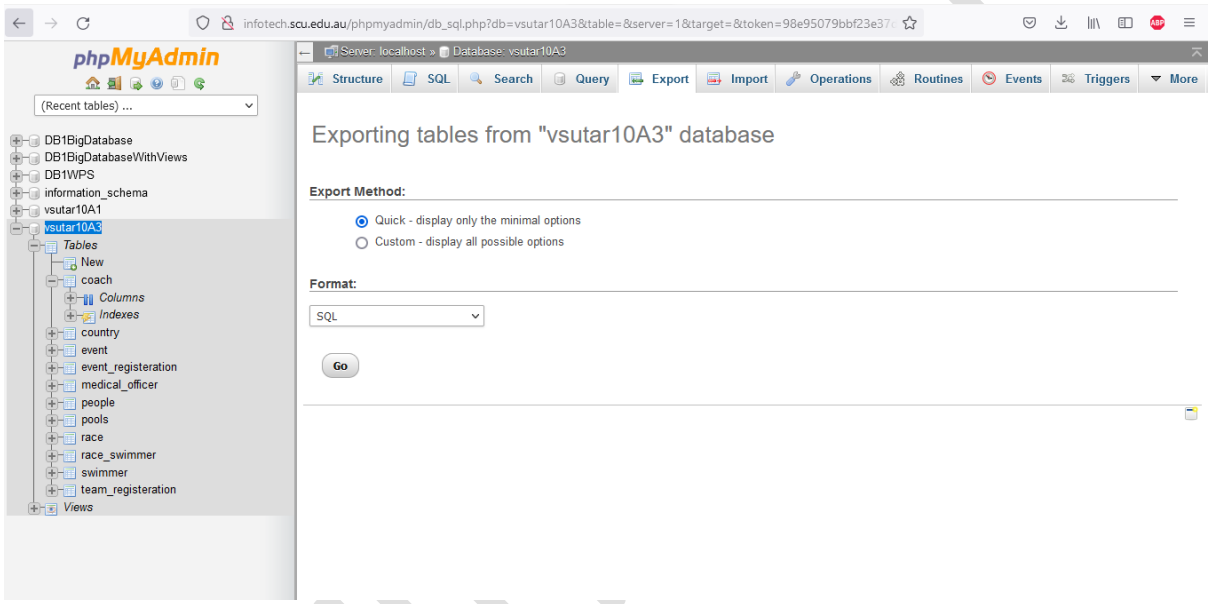


CREATE VIEW vw_RACE_SWIMMER AS
SELECT * FROM vw_TEAM_SWIMMER INNER JOIN race_swimmer on
race_swimmer.SwimmerID=vw_TEAM_SWIMMER.PersonID;



B.6 Export Script **1.5 MARKS**

Once your database is finalised, you must create an export script (.sql) to create a backup of your database, including all database structures, table definitions and data. Name this SQL script as yourusernameA3.sql e.g. aalaeiA3.sql.



B.7 Proof of Testing **1.5 MARKS**

The results of your queries should be presented in a Word document using the provided report template with a title page, page numbers etc. The query that you create for each of the parts below should be pasted into your report (not screenshots), followed by a screenshot of your results. Snipping Tool on Windows is useful for taking screenshots of query results.

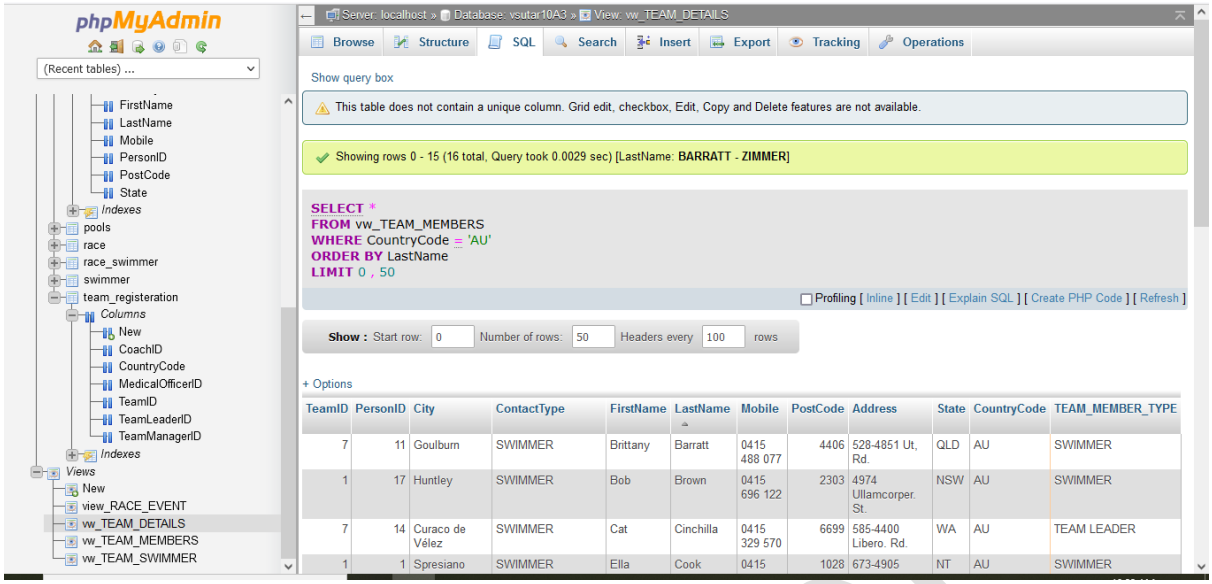
B.8 SQL Queries

Note: your result sets should not display any 'extra' columns – such as surrogate keys. Use surrogate keys in your progressive testing by all means, but your finished test results should not include surrogate keys unless specified in the question.

1. **Team Member list** **0.5 mark**

   Create a team list for one of the countries. Results should be sorted by last name of the team member, and should include all contact details and team member type (e.g. Swimmer, Coach, Medical Officer or other contact type).

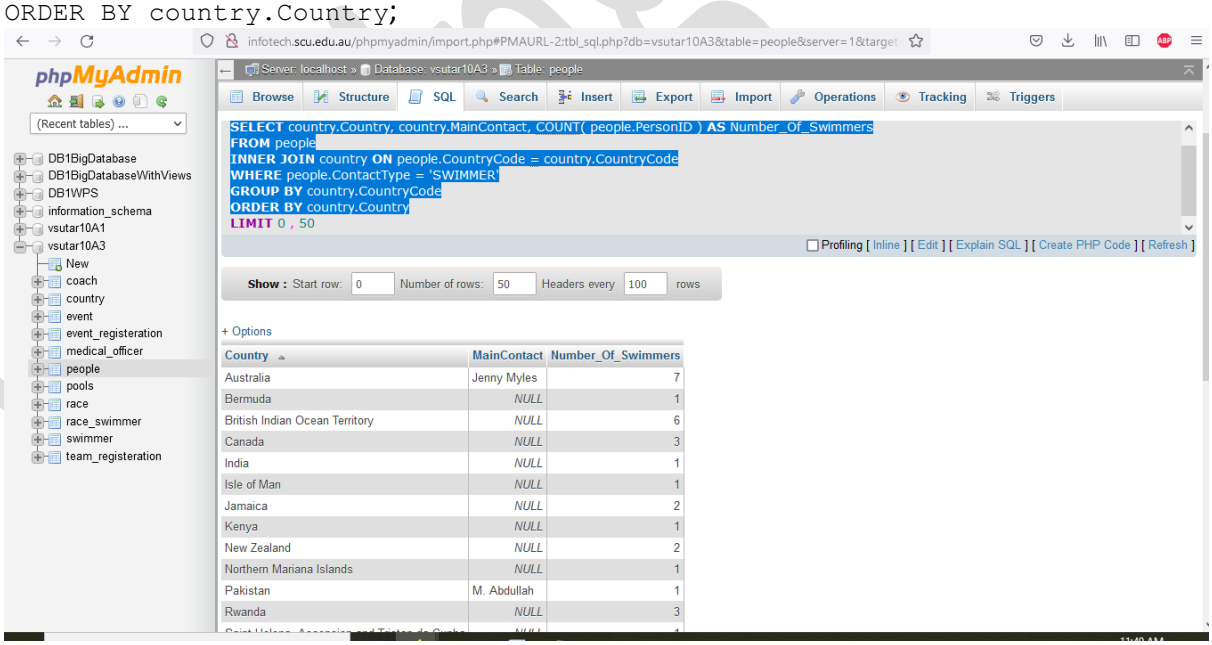
   SELECT * FROM vw_TEAM_MEMBERS WHERE CountryCode='AU' ORDER BY LastName;



2. **Country swimmer count** **0.5 mark**

Create a query that counts the number of swimmers for each country. Results should have the country name, main contact name, the number of swimmers, and be sorted by the country name.

```
SELECT country.Country, country.MainContact, COUNT( people.PersonID )
AS Number_Of_Swimmers
FROM people
INNER JOIN country ON people.CountryCode = country.CountryCode
WHERE people.ContactType = 'SWIMMER'
GROUP BY country.CountryCode
ORDER BY country.Country;
```



3. **Non-qualifying Swimmers** **0.5 mark**

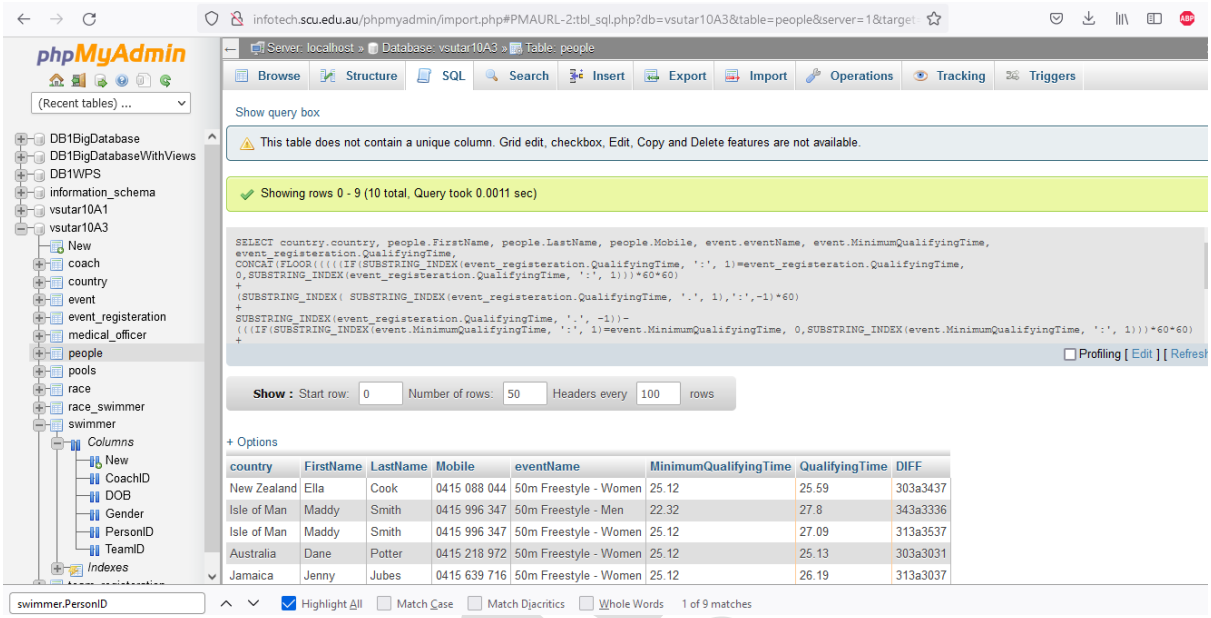
The Federation would like to identify swimmers that have not qualified for the events they have registered in. Create a query that selects these swimmers only, showing the country, swimmer name, email address, contact number along with the Event Name, minimum qualifying time, the swimmer's qualifying time and the difference between the times.

SELECT country.country, people.FirstName, people.LastName, people.Mobile, event.eventName, event.MinimumQualifyingTime, event_registeration.QualifyingTime,

```
CONCAT(FLOOR(((((IF(SUBSTRING_INDEX(event_registeration.QualifyingTime, ':',
1)=event_registeration.QualifyingTime,
0,SUBSTRING_INDEX(event_registeration.QualifyingTime, ':', 1)))*60*60)
+
(SUBSTRING_INDEX( SUBSTRING_INDEX(event_registeration.QualifyingTime, '.', 1),':',-
1)*60)
+
SUBSTRING_INDEX(event_registeration.QualifyingTime, '.', -1))-
(((IF(SUBSTRING_INDEX(event.MinimumQualifyingTime, ':',
1)=event.MinimumQualifyingTime,
0,SUBSTRING_INDEX(event.MinimumQualifyingTime, ':', 1)))*60*60)
+
(SUBSTRING_INDEX( SUBSTRING_INDEX(event.MinimumQualifyingTime, '.', 1),':',-
1)*60)
+
SUBSTRING_INDEX(event.MinimumQualifyingTime, '.', -1)))/60), ':',
LPAD(((((IF(SUBSTRING_INDEX(event_registeration.QualifyingTime, ':',
1)=event_registeration.QualifyingTime,
0,SUBSTRING_INDEX(event_registeration.QualifyingTime, ':', 1)))*60*60)
+
(SUBSTRING_INDEX( SUBSTRING_INDEX(event_registeration.QualifyingTime, '.', 1),':',-
1)*60)
+
SUBSTRING_INDEX(event_registeration.QualifyingTime, '.', -1))-
(((IF(SUBSTRING_INDEX(event.MinimumQualifyingTime, ':',
1)=event.MinimumQualifyingTime,
0,SUBSTRING_INDEX(event.MinimumQualifyingTime, ':', 1)))*60*60)
+
(SUBSTRING_INDEX( SUBSTRING_INDEX(event.MinimumQualifyingTime, '.', 1),':',-
1)*60)
+
SUBSTRING_INDEX(event.MinimumQualifyingTime, '.', -1)))%60, 2, 0)) AS DIFF

FROM country INNER JOIN people ON country.countryCode=people.countryCode
INNER JOIN swimmer ON people.PersonID=swimmer.PersonID INNER JOIN
event_registeration ON event_registeration.swimmerID=swimmer.PersonID INNER
JOIN event ON event.eventID=event_registeration.eventID WHERE
people.ContactType='swimmer'
AND
(((IF(SUBSTRING_INDEX(event.MinimumQualifyingTime, ':',
1)=event.MinimumQualifyingTime,
0,SUBSTRING_INDEX(event.MinimumQualifyingTime, ':', 1)))*60*60)
+
(SUBSTRING_INDEX( SUBSTRING_INDEX(event.MinimumQualifyingTime, '.', 1),':',-
1)*60)
+
SUBSTRING_INDEX(event.MinimumQualifyingTime, '.', -1))<
```

Write Sol

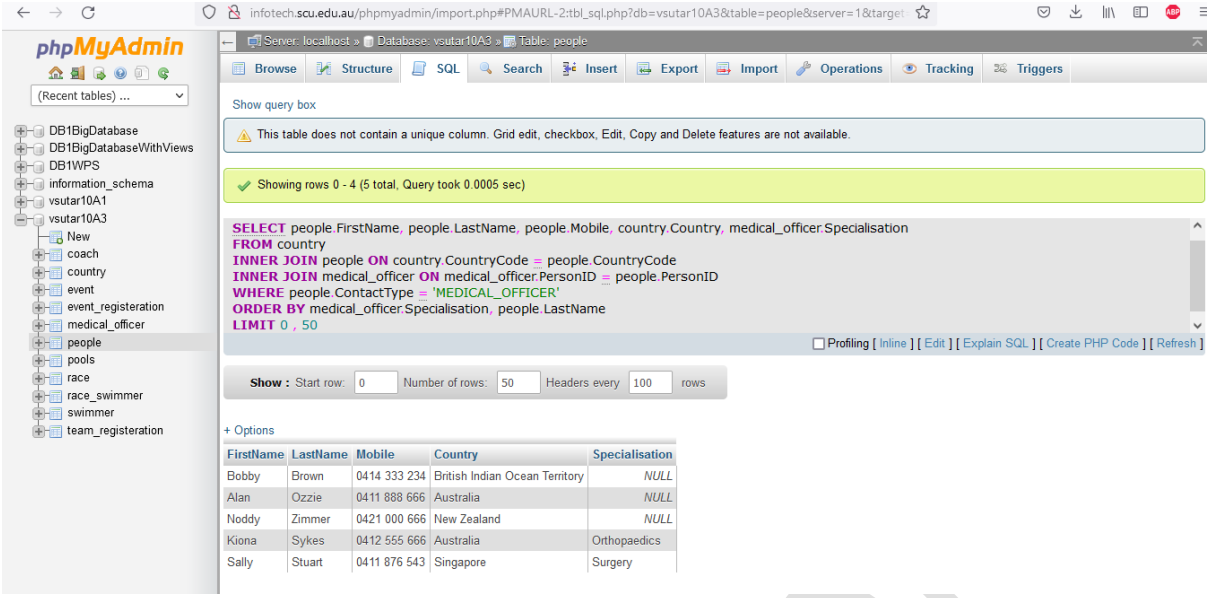
(((IF(SUBSTRING_INDEX(event_registeration.QualifyingTime, ':',
1)=event_registeration.QualifyingTime,
0,SUBSTRING_INDEX(event_registeration.QualifyingTime, ':', 1)))*60*60)
+
(SUBSTRING_INDEX(SUBSTRING_INDEX(event_registeration.QualifyingTime, '.', 1),':',-
1)*60)
+
SUBSTRING_INDEX(event_registeration.QualifyingTime, '.', -1));



4. **Medical Officer information** **0.5 marks**

Create a query that displays a list of medical officers, with the name, email and phone number, country and any specialisations, sorted by specialisation then last name.

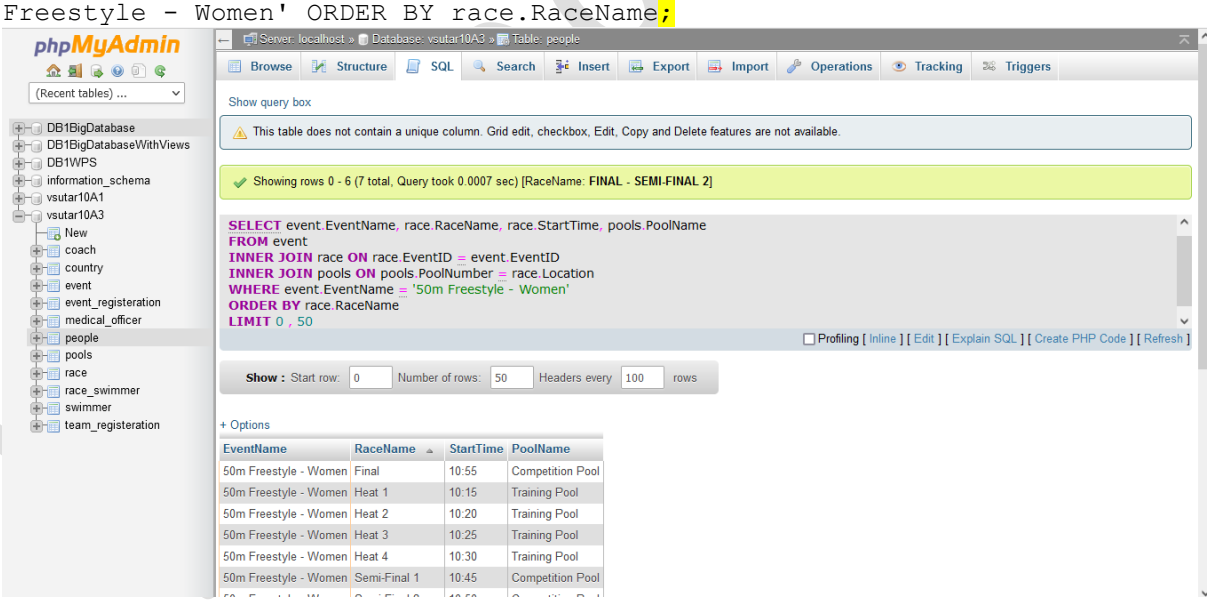
SELECT people.FirstName, people.LastName, people.Mobile, country.Country, medical_officer.Specialisation
FROM country INNER JOIN people ON country.CountryCode = people.CountryCode
INNER JOIN medical_officer ON medical_officer.PersonID = people.PersonID
WHERE people.ContactType = 'MEDICAL_OFFICER'
ORDER BY medical_officer.Specialisation, people.LastName;



5. **Race information** **1 marks**

Create a query that displays the list of races scheduled for one particular event. Races should be sorted by heat number, and display the event name, race name, starting time and location.

```
SELECT event.EventName, race.RaceName, race.StartTime, pools.PoolName
FROM event INNER JOIN race ON race.EventID = event.EventID INNER JOIN
pools ON pools.PoolNumber = race.Location WHERE event.EventName = '50m
Freestyle - Women' ORDER BY race.RaceName;
```

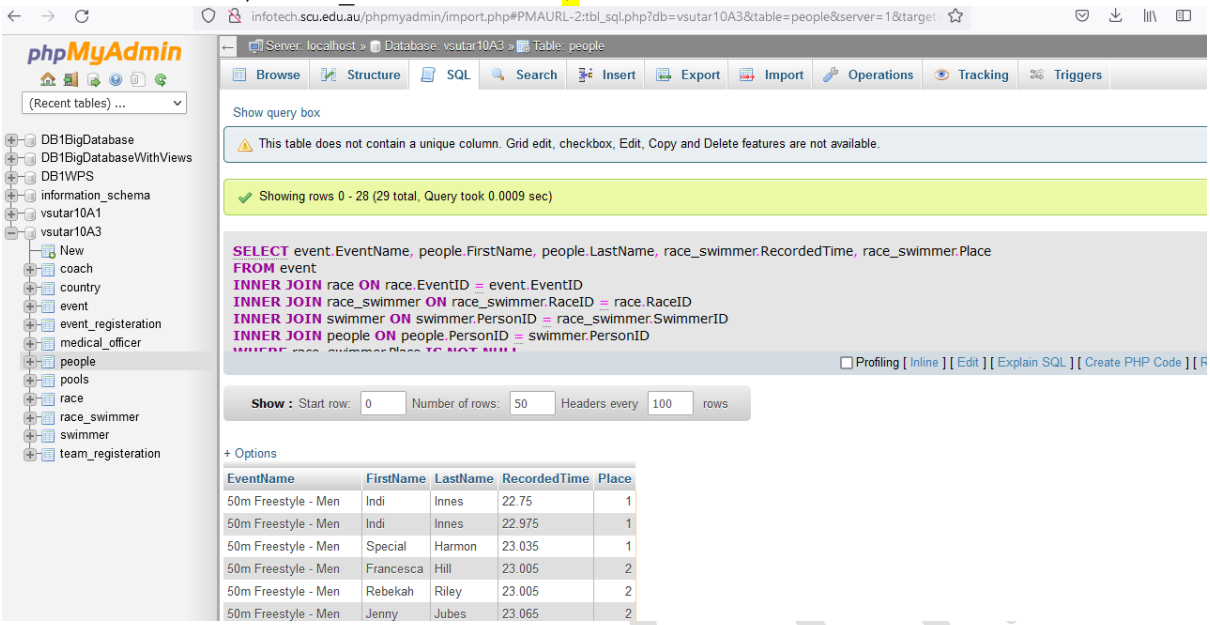


6. **Event Winner list** **1.5 marks**

Create a query to display the list of winners for each event, showing event name, swimmer name and recorded times for the Gold, Silver and Bronze winners, sorted by event name and by winning place.

```
SELECT event.EventName, people.FirstName, people.LastName,
race_swimmer.RecordedTime, race_swimmer.Place FROM event INNER JOIN
race ON race.EventID = event.EventID INNER JOIN race_swimmer ON
race_swimmer.RaceID = race.RaceID
INNER JOIN swimmer ON swimmer.PersonID = race_swimmer.SwimmerID INNER
JOIN people ON people.PersonID = swimmer.PersonID WHERE
```

```
race_swimmer.Place IS NOT NULL AND race_swimmer.Place <6 ORDER BY
event.EventName, race_swimmer.Place;
```

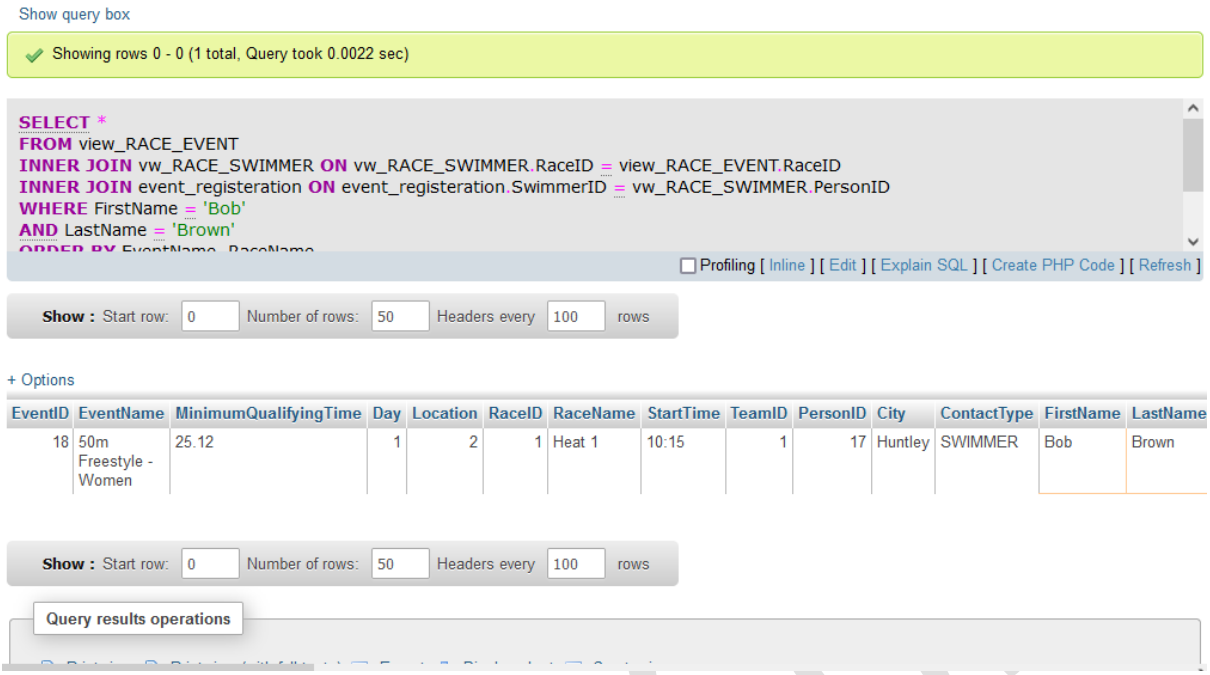


7. **Individual swimmer race results** **1.5 mark**

Create a query that displays a list of events for which Rebekah Riley (or use another name from your swimmer data) was registered and the races that they swam in. Results should be sorted by event name, race name, date and time and provide all recorded swim times, places and whether they won any medals.

```
SELECT *
FROM view_RACE_EVENT
INNER JOIN vw_RACE_SWIMMER ON vw_RACE_SWIMMER.RaceID =
view_RACE_EVENT.RaceID
INNER JOIN event_registeration ON event_registeration.SwimmerID =
vw_RACE_SWIMMER.PersonID
WHERE FirstName = 'Bob'
AND LastName = 'Brown'
ORDER BY EventName,RaceName,Day,StartTime;
```

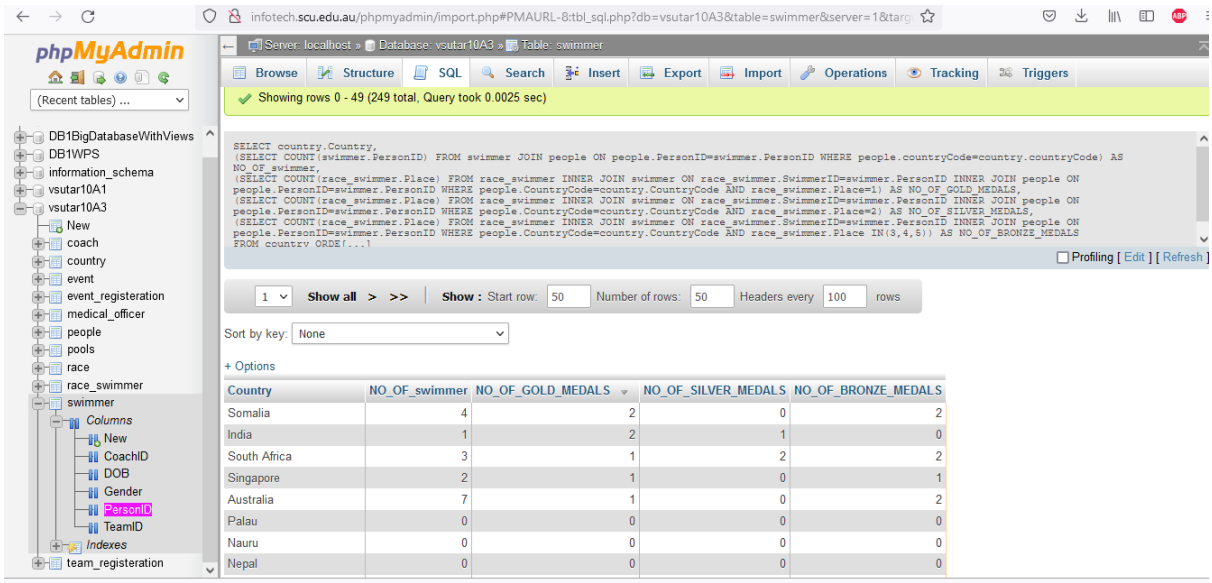
Writing Sc



8. **Medal Tally** **2 mark**

Create a query that displays each country with three calculated columns showing total number of swimmers and how many Gold, Silver and Bronze medals were won. Sort by number of Gold medals won.

```
SELECT country.Country,
(SELECT COUNT(swimmer.PersonID) FROM swimmer JOIN people ON
people.PersonID=swimmer.PersonID WHERE
people.countryCode=country.countryCode) AS NO_OF_swimmer,
(SELECT COUNT(race_swimmer.Place) FROM race_swimmer INNER JOIN swimmer ON
race_swimmer.SwimmerID=swimmer.PersonID  INNER JOIN people ON
people.PersonID=swimmer.PersonID WHERE
people.CountryCode=country.CountryCode AND race_swimmer.Place=1) AS
NO_OF_GOLD_MEDALS,
(SELECT COUNT(race_swimmer.Place) FROM race_swimmer INNER JOIN swimmer ON
race_swimmer.SwimmerID=swimmer.PersonID  INNER JOIN people ON
people.PersonID=swimmer.PersonID WHERE
people.CountryCode=country.CountryCode AND race_swimmer.Place=2) AS
NO_OF_SILVER_MEDALS,
(SELECT COUNT(race_swimmer.Place) FROM race_swimmer INNER JOIN swimmer ON
race_swimmer.SwimmerID=swimmer.PersonID  INNER JOIN people ON
people.PersonID=swimmer.PersonID WHERE
people.CountryCode=country.CountryCode AND race_swimmer.Place IN(3,4,5)) AS
NO_OF_BRONZE_MEDALS
FROM country ORDER BY NO_OF_GOLD_MEDALS DESC;
```


phpMyAdmin
(Recent tables) ...
DB1BigDatabaseWithViews
DB1WPS
information_schema
vsutar10A1
vsutar10A3
New
coach
country
event
event_registeration
medical_officer
people
pools
race
race_swimmer
swimmer
Columns
New
CoachID
DOB
Gender
PersonID
TeamID
Indexes
team_registeration
Server: localhost » Database: vsutar10A3 » Table: swimmer
Browse
Structure
SQL
Search
Insert
Export
Import
Operations
Tracking
Triggers
Showing rows 0 - 49 (249 total, Query took 0.0025 sec)
SELECT country.Country,
(SELECT COUNT(swimmer.PersonID) FROM swimmer JOIN people ON people.PersonID=swimmer.PersonID WHERE people.countryCode=country.countryCode) AS NO_OF_swimmer,
(SELECT COUNT(race_swimmer.Place) FROM race_swimmer INNER JOIN swimmer ON race_swimmer.SwimmerID=swimmer.PersonID INNER JOIN people ON people.PersonID=swimmer.PersonID WHERE people.CountryCode=country.CountryCode AND race_swimmer.Place=1) AS NO_OF_GOLD_MEDALS,
(SELECT COUNT(race_swimmer.Place) FROM race_swimmer INNER JOIN swimmer ON race_swimmer.SwimmerID=swimmer.PersonID INNER JOIN people ON people.PersonID=swimmer.PersonID WHERE people.CountryCode=country.CountryCode AND race_swimmer.Place=2) AS NO_OF_SILVER_MEDALS,
(SELECT COUNT(race_swimmer.Place) FROM race_swimmer INNER JOIN swimmer ON race_swimmer.SwimmerID=swimmer.PersonID INNER JOIN people ON people.PersonID=swimmer.PersonID WHERE people.CountryCode=country.CountryCode AND race_swimmer.Place IN(3,4,5)) AS NO_OF_BRONZE_MEDALS
FROM country ORDE[...]
Profiling [Edit] [Refresh]
Show all
Show : Start row: 50 Number of rows: 50 Headers every 100 rows
Sort by key: None
+ Options
Country NO_OF_swimmer NO_OF_GOLD_MEDALS NO_OF_SILVER_MEDALS NO_OF_BRONZE_MEDALS
Somalia 4 2 0 2
India 1 2 1 0
South Africa 3 1 2 2
Singapore 2 1 0 1
Australia 7 1 0 2
Palau 0 0 0 0
Nauru 0 0 0 0
Nepal 0 0 0 0

Writing So