LAB 5: More queries and web pages (SOLUTIONS)

Task 1: Queries

1. There may be a frequent need to look up packages by location and departure. Create (using SQL) a compound index called *IX_Packages_location_departure* on these two fields.

```
CREATE INDEX IX_Packages_location_departure ON Packages(location, departure)
```

It is decided that there is no need for this index. Using SQL, drop the index *IX_Packages_location_departure*

```
DROP INDEX Packages. IX_Packages_location_departure
```

2. In Lab 3 you found the package name and location for the tour with tourID = 3 Write this query using a join instead of a subquery.

```
SELECT packagename, location
FROM Packages,Tours
WHERE Packages.packageID = Tours.PackageID
AND tourID=3
```

3. List for every booking the first name and last name of the user and the number of adults and children booked

```
SELECT firstname, lastname, adults,children
FROM Bookings, Users
WHERE Bookings.username = Users.username
```

4. List for every booking for which tickets have not been sent the first name and last name of the user and the number of adults and children booked

```
SELECT firstname, lastname, adults, children
FROM Bookings, Users
WHERE Bookings.username = Users.username
AND status='tickets not sent'
```

5. List for every booking the tour departure date and the first name and last name of the user and the number of adults and children booked

```
SELECT departureDate, firstname, lastname, adults, children
FROM Bookings, Users, Tours
WHERE Bookings.username = Users.username
AND Tours.tourID = Bookings.tourID
```

6. List for every booking the package name, the tour departure date and the first name and last name of the user and the number of adults and children booked

```
SELECT packagename, departureDate, firstname, lastname, adults, children
FROM Bookings, Users, Tours, Packages
WHERE Bookings.username = Users.username
AND Tours.tourID = Bookings.tourID
AND Packages.packageID = Tours.packageID
```

7. You try to list the name, location and departure date for every tour using the following query. Run this query. What happens? Why?

```
SELECT packagename, location, departuredate
FROM Packages, Tours
```

Modify the query to give the result below.

```
SELECT packagename, location, departuredate
FROM Packages, Tours
WHERE Packages.packageID = Tours.packageID
```

8. List the tourID and departure date for every booking with at least one child. Write and run two versions of this query, one joining tables with WHERE and one using INNER JOIN.

```
SELECT Tours.tourID, departuredate
FROM Tours, Bookings
WHERE Tours.tourID = Bookings.tourID
AND children > 0
```

```
SELECT Tours.tourID, departuredate
FROM Tours
INNER JOIN Bookings ON Tours.tourID = Bookings.tourID
WHERE children > 0
```

9. List the booking ID, package name and total cost for every booking. *Hint: the total cost will be the (number of adults x adult price) + (number of children x child price)*

```
SELECT Bookings.bookingID, packagename, (adults*adultprice + children*childprice)
AS TotalCost
FROM Bookings, Tours, Packages
WHERE Bookings.tourID  = Tours.tourID
AND Packages.packageID = Tours.packageID
```

10. List the total cost of each user's bookings. *Hint: you will need to group by username*

```
SELECT username, SUM (adults*adultprice + children*childprice) AS SumTotalCost
FROM Bookings, Tours, Packages
WHERE Bookings.tourID  = Tours.tourID
AND Packages.packageID = Tours.packageID
GROUP BY username
```

11. Find the number of bookings made for each location

```
SELECT location, Count(Bookings.bookingID) AS [NoOfBookings]
FROM Bookings, Tours, Packages
WHERE Bookings.tourID  = Tours.tourID
AND Packages.packageID = Tours.packageID
GROUP BY location
```

12. Increase the prices of all packages by 10%.

```
UPDATE Packages
SET
adultprice = adultprice * 1.1,
childprice = childprice * 1.1
```

13. Change the password for mpolo to Pa$$w0rd

```
UPDATE Users
SET
password = 'Pa$$w0rd'
WHERE username = 'mpolo'
```

14. Delete the tour with tourID 5.

```
DELETE FROM Tours
WHERE tourID = 5
```

This query shouldn't work! Why not?

Use ALTER TABLE statements to:

- **Drop the foreign key relationship between Tours and Bookings**
- **Add a new foreign key relationship between Tours and Bookings so that when you delete a tour you delete related bookings**

Run your query again and note changes to both tables.

```
ALTE|R TABLE Bookings DROP CONSTRAINT FK_Bookings_Tours

ALTER TABLE Bookings ADD CONSTRAINT FK_ Bookings_Tours
FOREIGN KEY(tourID) REFERENCES Tours(tourID)
```

Task 2: Creating web pages to view data

cartypes.cshtml

```
@{
    var db = Database.Open("gcucars");
    var selectQueryString = "SELECT * FROM Cartypes";
 }

<!DOCTYPE html>
<html>
  <head>
    <title>GCUCars Car Types</title>
  </head>
  <body>
    <h1>GCUCars Car Types</h1>
    <table>
        <thead>
            <tr>
                <th>Description</th>
                <th>Daily cost</th>
                <th>CDW daily cost</th>
                <th>Models</th>
            </tr>
        </thead>
        <tbody>
            @foreach (var row in db.Query(selectQueryString)){
             <tr>
                  <td>@row.description</td>
                  <td>£@row.dailyCost</td>
                  <td>£@row.CDWDailyCost</td>
                  <td><a href="carmodels.cshtml?description=
                      @row.description">models</a></td>
             </tr>
            }
        </tbody>
    </table>
    <p>The SQL query which retrieved the data for this page
       was:<br/>
     @selectQueryString</p>
  </body>
</html>
```

carmodels.cshtml

```
@{
    var type = Request["description"];
    var db = Database.Open("gcucars");
    var selectQueryString =
        "SELECT * FROM CarModels WHERE carType='" + type + "'";
 }

<!DOCTYPE html>
<html>
  <head>
    <title>GCUCars Car Models: @type</title>
  </head>
  <body>
    <h1>GCUCars Car Models: @type</h1>
    <table>
        <thead>
            <tr>
                <th>Make</th>
                <th>Model</th>
                <th>Engine Size</th>
                <th>Engine Type</th>
            </tr>
        </thead>
        <tbody>
            @foreach (var row in db.Query(selectQueryString)){
             <tr>
                  <td>@row.make</td>
                  <td>@row.model</td>
                  <td>@row.engineSize</td>
                  <td>@row.engineType</td>
             </tr>
            }
        </tbody>
    </table>
    <p><a href="cartypes.cshtml">All types</a></p>
    <p>The SQL query which retrieved the data for this page
     was:<br/>
     @selectQueryString</p>
  </body>
</html>
```