

LAB 5: More queries and forms (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 *idx_location_departure* on these two fields.

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

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

```
DROP INDEX idx_location_departure ON Packages
```

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 name, 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
```

firstname	lastname	adults	children
Marco	Polo	2	2
Marco	Polo	1	0
Marco	Polo	2	2
Vasco	daGama	4	0
Vasco	daGama	2	0
Ferdinand	Magellan	2	3

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'
```

firstname	lastname	adults	children
Marco	Polo	1	0
Marco	Polo	2	2
Vasco	daGama	2	0
Ferdinand	Magellan	2	3

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
```

departureDate	firstname	lastname	adults	children
01/03/2008	Marco	Polo	2	2
05/06/2008	Marco	Polo	1	0
01/08/2008	Marco	Polo	2	2
01/03/2008	Vasco	daGama	4	0
02/09/2008	Vasco	daGama	2	0
01/06/2008	Ferdinand	Magellan	2	3

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 name, 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
```

name	departureDate	firstname	lastname	adults	children
Western Adventure	01/03/2008	Marco	Polo	2	2
Roof of the World Explorer	05/06/2008	Marco	Polo	1	0
Amazon & Inca Adventure	01/08/2008	Marco	Polo	2	2
Roof of the World Explorer	01/03/2008	Vasco	daGama	4	0
Reef and Outback Adventure	02/09/2008	Vasco	daGama	2	0
Trans-Siberian Express	01/06/2008	Ferdinand	Magellan	2	3

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 name, location, departuredate
FROM Packages, Tours
```

Modify the query to give the result below.

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

name	location	departuredate
Western Adventure	USA	01/03/2008
Western Adventure	USA	05/06/2008
Western Adventure	USA	02/09/2008
Roof of the World Explorer	Asia	01/03/2008
Roof of the World Explorer	Asia	05/06/2008
Alpine Action	Europe	01/03/2008
Reef and Outback Adventure	Australia	01/03/2008
Reef and Outback Adventure	Australia	02/09/2008
Trans-Siberian Express	Asia	01/03/2008
Trans-Siberian Express	Asia	01/06/2008
Trans-Siberian Express	Asia	01/08/2008
Borneo Explorer	Asia	08/03/2008
Amazon & Inca Adventure	South America	01/02/2008
Amazon & Inca Adventure	South America	01/08/2008
Patagonia Trek	South America	01/05/2008
Colorado Winter Adventure	USA	01/02/2008
Colorado Winter Adventure	USA	01/03/2008
Raft the Grand Canyon	USA	01/05/2008
Raft the Grand Canyon	USA	01/06/2008
Raft the Grand Canyon	USA	01/07/2008
Raft the Grand Canyon	USA	01/08/2008
Rising Sun Explorer	Asia	01/07/2008

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
```

tourID	departuredate
1	01/03/2008
14	01/08/2008
10	01/06/2008

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

bookingID	name	TotalCost
1	Western Adventure	£4,996.00
2	Roof of the World Explorer	£1,599.00
3	Amazon & Inca Adventure	£6,996.00
4	Roof of the World Explorer	£6,396.00
5	Reef and Outback Adventure	£4,398.00
8	Trans-Siberian Express	£4,795.00

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
```

username	SumTotalCost
ferdy	£4,795.00
mpolo	£13,591.00
vdagama	£10,794.00

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
```

location	NoOfBookings
Asia	3
Australia	1
South America	1
USA	1

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. Create a parameter query which you can run to change the password for any user. Your query should prompt for the username first, and then the password.

```
PARAMETERS [which user?] Text, [new password] Text;
UPDATE Users
SET
password = [new password]
WHERE username = [which user?]
```

15. Delete the tour with tourID 5.

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

This query shouldn't work! Why not?

Edit the relationship between Tours and Bookings so that when you delete a tour you delete related bookings (you'll need to do this with the Relationships window, not with SQL). Run your query again and note changes to both tables.

Task 3: Using a query to help normalisation

3. Devise and run a query which inserts the correct data into *Skills*, and drop the *hourlyRate* field from *ConsultantSkills*. You should NOT type any data into the table manually. Copy and paste the SQL you use into your Word document under the heading *Task 3 part 2*.

```
INSERT INTO Skills  
SELECT DISTINCT skill AS skill, hourlyRate AS hourlyRate  
FROM ConsultantSkills;
```