Exercise 1. [up to 8 pt] Remember relational schema for Library from Practice 3/Ex 1:

**Reader( ID, LastName, FirstName, Address, BirthDate)  
Book ( ISBN, Title, Author, PagesNum, PubYear, PubName)  
Publisher ( PubName, PubKind)  
Category ( CategoryName, ParentCat)  
Copy ( ISBN, CopyNumber, Shelf, Position)  
Borrowing ( ReaderNr, ISBN, CopyNumber, ReturnDate)  
BookCat ( ISBN, CategoryName )**

Write SQL SELECTs (or relational algebra expressions) for the following questions:

1. What are the last names of the readers in Moscow?
2. Which books (author, title) are published by scientific and reference books publishers (pubKind is Science or Reference)?
3. Which books (author, title) are borrowed by Ivan Ivanov?
4. Which books (ISBN) from the category "Mountains" doesn’t belong to category "Travel"? Subcategories do not have to be taken into account!
5. Which readers (LastName, FirstName) have borrowed books that were returned?
6. Which readers (LastName, FirstName) have borrowed at least one book, borrowed also by Ivan Ivanov (don’t include Ivan Ivanov in the result)?

Решение:

1.

SELECT LastName FROM Reader WHERE instr(Address, 'Moscow') > 0;

2.

SELECT DISTINCT Author, Title FROM Book  
JOIN main.Publisher P on Book.PubName = P.PubName  
WHERE instr(P.PubKind , 'Scientific') > 0 OR instr(P.PubKind, 'Reference') > 0;

3.

SELECT DISTINCT Author, Title FROM Book  
JOIN Borrowing B on Book.ISBN = B.ISBN  
JOIN main.Reader R on B.ReaderNr = R.ID  
WHERE R.LastName = 'Ivanov' AND R.FirstName = 'Ivan';

4.

SELECT DISTINCT Book.ISBN  
FROM Book  
 JOIN BookCat BC on Book.ISBN = BC.ISBN  
WHERE BC.CategoryName = 'Mountains'  
 AND Book.ISBN NOT IN (  
 SELECT DISTINCT Book.ISBN  
 FROM Book  
 JOIN BookCat BC on Book.ISBN = BC.ISBN  
 WHERE BC.CategoryName = 'Travel');

5.

SELECT DISTINCT LastName, FirstName  
FROM Reader  
 JOIN Borrowing B on ID = B.ReaderNr  
WHERE B.ISBN IN (  
 SELECT Book.ISBN  
 FROM Book  
 JOIN Borrowing B2 on Book.ISBN = B2.ISBN  
 WHERE B2.ReturnDate < B.ReturnDate);

6.

SELECT DISTINCT LastName, FirstName  
FROM Reader  
 JOIN Borrowing B on ID = B.ReaderNr  
WHERE B.ISBN IN (  
 SELECT Book.ISBN  
 FROM Book  
 JOIN Borrowing B2 on Book.ISBN = B2.ISBN  
 JOIN Reader R2 on R2.ID = B2.ReaderNr  
 WHERE R2.FirstName = 'Ivan'  
 AND R2.LastName = 'Ivanov'  
)  
 AND NOT (Reader.LastName = 'Ivanov'  
 AND Reader.FirstName = 'Ivan');

Exercise 2.\*\* [1 pt] Remember relational schema for Trains from Practice 3/Exercise 4a

**City ( Name, Region )  
Station ( Name, #Tracks, CityName, Region )  
Train ( TrainNr, Length, StartStationName, EndStationName )  
Connection ( TrainNr, FromStation, ToStation, Departure, Arrival)**

Assume that the relation "Connection" already **contains the transitive closure**. That is when train 101 leaves from Moscow to Sankt-Petersburg via Tver, contains the relation tuples for Moscow->Tver, Tver-Sankt-Petersburg and Moscow-> Sankt-Petersburg. Formulate the following queries in relational algebra:

1. Find all direct connections from Moscow to Tver.
2. Find all multi-segment routes having precisely single-day transfer from Moscow to Sankt-Petersburg (first departure and final arrival should go on the same date). You can apply the function DAY () to attributes Departure and Arrival to determine the date.
3. What will change in expressions for a) and b), if the relation "Connection" has no additional tuples for the transitive closure contains, so the multi-segment route Moscow->Tver-> Sankt-Petersburg, contains only tuples for Moscow->Tver and Tver-Sankt-Petersburg?

Решение:

1.

SELECT Train.TrainNr  
FROM Train  
 JOIN Connection C on Train.TrainNr = C.TrainNr  
 JOIN Station fs ON C.FromStation = fs.Name  
 JOIN Station ts ON C.ToStation = ts.Name  
WHERE fs.CityName = 'Moscow'  
 AND ts.CityName = 'Tver'  
 AND NOT EXISTS(  
 SELECT \*  
 FROM Connection c2  
 WHERE c2.TrainNr = C.TrainNr  
 AND (C.FromStation = c2.FromStation AND C.ToStation != c2.ToStation AND NOT EXISTS(  
 SELECT \*  
 FROM Connection c3  
 WHERE c3.TrainNr = c2.TrainNr  
 AND (c2.ToStation = c3.ToStation AND C.ToStation = c3.FromStation)  
 ))  
 OR (C.FromStation != c2.FromStation AND C.ToStation = c2.ToStation AND NOT EXISTS(  
 SELECT \*  
 FROM Connection c3  
 WHERE c3.TrainNr = c2.TrainNr  
 AND (c3.FromStation = C.FromStation AND C.ToStation = c3.ToStation)  
 ))  
 );

2.

SELECT Train.TrainNr  
FROM Train  
 JOIN Connection C on Train.TrainNr = C.TrainNr  
 JOIN Station fs ON C.FromStation = fs.Name  
 JOIN Station ts ON C.ToStation = ts.Name  
WHERE fs.CityName = 'Moscow'  
 AND ts.CityName = 'Sankt-Petersburg'  
 AND DATE(C.Departure) = DATE(C.Arrival)  
 AND EXISTS(  
 SELECT \*  
 FROM Connection c2  
 WHERE c2.TrainNr = C.TrainNr  
 AND (c2.ToStation = C.FromStation  
 OR c2.FromStation = C.ToStation  
 OR (c2.FromStation = C.FromStation AND NOT c2.ToStation = C.ToStation)  
 OR (NOT c2.FromStation = C.FromStation AND c2.ToStation = C.ToStation))  
 );

3.

Так как мы не умеем выражать транзитивное замыкание средствами реляционной алгебры, если его нет, точно его создать с помощью стандартных запросов SQL нельзя. В таких ситуациях можно делать джойн таблицы Connection с самой собой, но делать их придется для каждого случая конкретное требуемое число раз.

Exercise 3.\* [1pt] Represent outer join as relational algebra expression using only basic operations (select, project, cartesian, rename, union, minus)

Решение:

Если мы хотим сделать outer join двух таблиц A и B по общим столбцам, то нам требуется сделать джойн совпадающих по столбцам строк и заполнить недостающие значения null-ами.

Для этого нам нужно получить результат

InnerJoin = πa1 ... am, b1 ... bn (σ∀i A.xi = B.xi(A×B)),

а затем добавить к нему то, что не получилось спроецировать из столбцов одной и другой таблицы (условно левой и правой):

remainsR = (πBInnerJoin - B) x {(null, null, ... null)}

remainsL = (A - πAInnerJoin) x {(null, null, ... null)}

И далее их объединить.

Ответ: InnerJoin ∪ remainsL ∪ remainsR