Shilova Ksenia, 196

Task 1

Book({ISBN:string, title:string, year:int , author:string, pagesNum:int, publisherName: string})

Copy({number:int, shelfPosition:int, bookISBN:int})

Publisher({name:string, address:string})

Category({name:string, superCategoryName:string})

Book_Category_belongs ({categoryName:string, bookISBN:int})

Reader({id:int, firstName:string, lastName:string, address:string, birthday:date})

Borrowing({readerId:int, copyNum:int, copyISBN:int, returnDate:date})

А) Какие фамилии читателей в Москве?

```
select lastName
from Reader
where “Москва” in address
```

Б) Какие книги (author, title) брал Иван Иванов?

```
select distinct author, title
from Borrowing borr
    join Reader r on borr.readerId = r.id
    join Book b on borr.copyISBN = b.ISBN
 where r.firstName = “Иван” and r.lastName = “Иванов”
```

**Соединяем таблицы Book, Reader, Borrowing по соответсвующим полям, затем выбираем Ивана Иванова и делаем distinct, так как тогда не будет дубликатов. Author, title уникальные имена полей, можно брать их без обращения к таблице.*

**Второй способ: заменить первую строку. Тогда будет весь список книг, даже если они повторяются. Если удаление дубликатов не обязательно, то лучше делать без distinct.*

```
select author, title
```

В) Какие книги (isbn) из категории «Горы» не относятся к категории «Путешествия»? Подскатегории не обязательно принимать во внимание

```
select bc.ISBN
from  Book_Categoty_belongs bc
```

```
where bc.categoryName = “Горы”

minus

select bc.ISBN

from  Book_Categoty_belongs bc

where bc.categoryName = “Птешествия”
```

Г) Какие читатели (LastName, FirstName) вернули копию книгу?

```
select lastName, firstName

from Borrowing borr

    join Reader r on borr.readerId = r.id

 where currentDate > borr.returnDate
```

**Предполагается, что currentDate известна*

Д) Какие читатели (LastName, FirstName) брали хотя бы одну книгу (не копию), которую брал также Иван Иванов (не включайте Ивана Иванова в результат)

*Книги, которые брал Иван Иванов:

```
select ISBN

from Borrowing borr

    join Reader r on borr.readerId = r.id

 where r.firstName = “Иван” and r.lastName = “Иванов”
```

*Выбираем тех читателей, которые не являются Иваном Ивановым и брали хотя бы одну книгу из первого запроса

```
select distinct lastName, firstName

from Borrowing borr

    join Reader r on borr.readerId = r.id

 where not r.firstName = “Иван” or not r.lastName = “Иванов” and

        borr.ISBN in (select ISBN

                    from Borrowing borr

                        join Reader r on borr.readerId = r.id
```

where r.firstName = “Иван” **and** r.lastName = “Иванов”

)

Task 2

City({region:string, name:string})

Station({name:string, cityRegion:string, cityName:string, #tracks:int })

Train({TrainNr:int, length:int, startStationName:string, endStationName:string})

Connection({fromStationName:string, trainNr:int, arrival:time, departure:time, toStationName:string})

**Будем считать (как в лекции), что $\Pi_{a1,a2,\dots}Table$ это выбор столбцов из таблицы, а условие обозначается как $\sigma_{condition}Table$ это выбор строк по условию. ^ - and. × - join.*

а) Найдите все прямые рейсы из Москвы в Тверь.

**Нужно взять те номера поездов (номер маршрута = рейс), которые есть в Connection вместе с fromStation в городе Москва и toStation в городе Тверь. Считается, что прямой рейс – это тот рейс, при котором не надо делать пересадку. То есть если номер поезда одинаковый и есть остановки, то это все равно прямой рейс.*

$$\Pi_{\text{trainNr,fromStationName,toStationName,arrival,departure,cytiNameFrom,cityName as cityNameTo}}($$

$$\sigma_{\text{Connection.toStationName=Station.name}}($$

$$\Pi_{\text{trainNr,fromStationName,toStationName,arrival,departure,cityName as cityNameFrom}}($$

$$\sigma_{\text{Connection.fromStationName=Station.name}}(\text{Connection} \times \text{Station})) \times Station))$$

**То есть, аналогично пункту 2 (там расписано подробно) берем и находим сначала город для станции fromStationName, а затем для станции toStationName. Присоединяем эти два столбца (переименовывая в cityNameFrom, cityNameTo) к таблице Connection.*

Затем находим те, для которых cityNameFrom = Москва, а cityNameTo = Тверь

Итоговый ответ:

$$\Pi_{trainNr}(\sigma_{cityNameFrom="Москва" \wedge cityNameTo="Тверь"}($$

$$\Pi_{trainNr,fromStationName,toStationName,arrival,departure,cytiNameFrom,cityName\ as\ cityNameTo}($$

$$\sigma_{Connection.toStationName=Station.name}($$

$$\Pi_{trainNr,fromStationName,toStationName,arrival,departure,cityName\ as\ cityNameFrom}($$

$$\sigma_{Connection.fromStationName=Station.name}(Connection \times Station)\ ) \times Station))$$

$$)\ )$$

б) Найдите все многосегментные маршруты, имеющие точно однодневный трансфер из Москвы в Санкт-Петербург (первое отправление и прибытие в конечную точку должны быть в одну и ту же дату). Вы можете применить функцию DAY () к атрибутам Departure и Arrival, чтобы определить дату.

**Надо найти такие connection, чтобы fromStation была в Москве и toStation была в СПб. Затем нужно взять те их них, для которых у поездов есть еще какие-либо остановки, кроме start и end. То есть взять только многосегментные.*

Затем нужно взять у этих Connection даты и найти те из них, где даты отправки и прибытия не отличаются.

Сделаем по действиям и затем объединим выражения:

1) Соединим таблицы Station и Connection, чтобы легко узнавать город станции FROM. Добавим переименование столбца cityName as cityNameFrom.

$$\Pi_{trainNr,fromStationName,toStationName,arrival,departure,cityName\ as\ cityNameFrom}($$

$$\sigma_{Connection.fromStationName=Station.name}(Connection \times Station)\ )$$

2) Теперь сделаем то же самое, чтобы узнать город станции ToStation.

$$\Pi_{trainNr,fromStationName,toStationName,arrival,departure,cytiNameFrom,cityName\ as\ cityNameTo}($$

$$\sigma_{Connection.toStationName=Station.name}($$

$$\Pi_{trainNr,fromStationName,toStationName,arrival,departure,cityName\ as\ cityNameFrom}($$

$$\sigma_{Connection.fromStationName=Station.name}(Connection \times Station)\ )$$

$$\times Station))$$

**В итоге получили таблицу из столбцов trainNr, fromStationName, toStationName, arrival, departure, cityNameFrom, cityNameTo.*

3) Теперь возьмем из этой таблицы только те строки, где cityNameFrom = Москва, а cityNameTo = Спб.

$$\sigma_{cityNameFrom="Москва" \wedge cityNameTo="Санкт-Петербург"}($$

$$\Pi_{\text{trainNr,fromStationName,toStationName,arrival,departure,cytiNameFrom,cityName as cityNameTo}}($$

$$\sigma_{\text{Connection.toStationName=Station.name}}($$

$$\Pi_{\text{trainNr,fromStationName,toStationName,arrival,departure,cityName as cityNameFrom}}($$

$$\sigma_{\text{Connection.fromStationName=Station.name}}(\text{Connection} \times \text{Station})) \times \text{Station})))$$

4) Теперь возьмем только те строки, для которых этот маршрут является многосегментным. То есть, должны быть еще и другие коннекшены для каждого поезда, кроме тех, которые уже нашли (мск – спб).
Обозначим предыдущую полученную таблицу за (…), чтобы не переписывать.
Тогда:
Условие того, что у поезда только две остановки start и end:

$$\text{COUNT}(\sigma_{\text{Train.TrainNr=(…).TrainNr}}(\text{Train} \times (\dots))) = \text{COUNT}(\text{Train})$$

Тогда, вычитаем все такие картежи, которые удовлетворяют этому условию

$$(\dots) - \sigma_{\text{COUNT}(\sigma_{\text{Train.TrainNr=Connection.TrainNr}}(\text{Train}\times\text{Connection})=\text{COUNT}(\text{Train}))} (\dots)$$

5) И теперь осталось взять только те поезда, где DAY(arrival) == DAY(departure)

$$\Pi_{trainNr}(\sigma_{\text{DAY(arrival)=DAY(departure)}}((\dots) - \sigma_{\text{COUNT}\left(\sigma_{\text{Train.TrainNr=(…).TrainNr}}(\text{Train}\times(\dots))=\text{COUNT}(\text{Train})\right)} (\dots)))$$

Итоговый ответ:

$$\Pi_{trainNr}(\sigma_{\text{DAY(arrival)=DAY(departure)}}((\dots) - \sigma_{\text{COUNT}\left(\sigma_{\text{Train.TrainNr=(…).TrainNr}}(\text{Train}\times(\dots))=\text{COUNT}(\text{Train})\right)} (\dots)))$$

где

$$(\dots) = \sigma_{\text{cityNameFrom="Москва" ^ cityNameTo="Санкт–Петербург"}}($$

$$\Pi_{\text{trainNr,fromStationName,toStationName,arrival,departure,cytiNameFrom,cityName as cityNameTo}}($$

$$\sigma_{\text{Connection.toStationName=Station.name}}($$

$$\Pi_{\text{trainNr,fromStationName,toStationName,arrival,departure,cityName as cityNameFrom}}($$

$$\sigma_{\text{Connection.fromStationName=Station.name}}(\text{Connection} \times \text{Station})) \times \text{Station})))$$

в) Что изменится в выражениях для а) и б), если отношение "Connection" не содержит дополнительных кортежей для транзитивного замыкания, поэтому многосегментный маршрут Москва-> Тверь-> Санкт-Петербург содержит только кортежи Москва-> Тверь и Тверь-Санкт-Петербург?

Итоговый ответ:

Тогда, пункт а) можно записать следующим образом:

взять все connection для всех поездов (декартово произведение, то есть ×) и если среди них есть одна запись, для которой fromStation в Москве и хотя бы одна, для которой toStation в Твери, то этот маршрут удовлетворяет условиям прямого рейса.

Пункт б) следующим образом:

Как в пункте а) находим рейсы из Москвы и СПб, только для записи, для которой fromStation в Москве находим DAY(arrival), Затем находим DAY(arrival) для той строки, для которой toStation в СПб. Берем только те номера поездов, для которых эти дни совпали. Затем вычитаем из них те строки, для поездов из которых выполняется то же условие, что и было (про многосегментность).

Task 3

Представьте внешнее объединение (outer join) в виде выражения реляционной алгебры с использованием только базовых операций (select, project, cartesian, rename, union, minus)

Пусть нужно объединить отношение R1 с доменами A, B (столбцами) и R2 с B, C.

Сначала определим NaturalJoin:

$$NJ = \Pi_{A,B,C}(\sigma_{R1.B=R2.B}\,(\mathrm{R1} \times \mathrm{R2}))$$

Записи, которые не попали в NJ из R1 дозаполняем null:

$$\Big(R1 - \Pi_{A,B}(NJ)\Big) \times ((null, null), \dots (null, null))$$

Аналогично с R2:

$$((null, null), \dots (null, null)) \times \Big(R2 - \Pi_{B,C}(NJ)\Big)$$

Объединяем все эти записи:

$$\Big(R1 - \Pi_{A,B}(NJ)\Big) \times \big((null, null), \dots (null, null)\big) \cup \big((null, null), \dots (null, null)\big) \times \Big(R2 - \Pi_{B,C}(NJ)\Big) \cup NJ$$