

31.10.2023

Übungsblatt 4 – Lösungsvorschlag


Hinweis



Greifen Sie gerne auf das webbasierte Übungstool für relationale Algebra *RelaX^a* zurück, um auf den gegebenen Relationen Relationenalgebra-Operationen auszuführen und den zu diesen zugehörigen Operatorbaum zu generieren.

^a<https://dbis-uibk.github.io/relax>

Diskussionsteil (im PS zu lösen; keine Abgabe nötig)

- a)  Gegeben sei ein Relationenmodell mit Relationenschemata Member (Id, Age, Place, RegistrationDate)¹ und Chapter (Name, Location)² mit folgenden Tupeln:

Member				
Id	Name	Age	Place	RegistrationDate
143	Schmidt, M.	20	Bremen	2023-07-14
145	Huber, Chr.	21	Augsburg	2019-03-29
146	Abele, I.	22	Senden	2018-09-05
149	Kircher, B.	23	Bochum	2019-12-18
155	Meier, W.	24	Stuttgart	2023-04-02
171	Möller, H.	25	Innsbruck	2021-10-21
173	Schulze, B.	26	Stuttgart	2015-08-09
177	Mons, F.	26	Essen	2022-02-15
185	Meier, K.	27	Heidelberg	2016-06-26
187	Karstens, L.	27	Hamburg	2017-11-08
194	Gerstner, M.	28	Innsbruck	2023-05-30

Chapter	
Name	Location
Thunderstrike Athletics	Augsburg
Phoenix Cyclones FC	Bregenz
Starfire Racquet Club	Munich
Alpine Ridge Ski Club	Innsbruck
Avalanche Warriors Hockey	Heidelberg
Galaxy Runners Track & Field	Innsbruck

¹dom(ID) = dom(Age) = Integer, dom(Place) = String und dom(RegistrationDate) = Datum im Format YYYY-MM-DD

²dom(Name) = dom(Location) = String

Hinweis



Unter

dbis-uibk.github.io/relax/calc/gist/a931db36e5273ca284714dd81c9109b2

können Sie die Datenbank „Sports Club“^a in RelaX öffnen, um auf ihren Relationen Relationenalgebra-Operationen auszuführen.

^ahttps://gist.github.com/antstei/a931db36e5273ca284714dd81c9109b2#file-sports_club_description-md

a) Bestimmen Sie $\sigma_{\text{Age} < 25}(\text{Member})$. Geben Sie dazu

- das von Ihnen zunächst ohne Rückgriff auf RelaX berechnete Ergebnis, das Sie anschließend mithilfe von RelaX überprüfen, und
- eine Beschreibung von diesem in eigenen Worten an.

Lösung



i. Member.Id	Member.Name	Member.Age	Member.Place	Member.RegistrationDate
143	'Schmidt, M.'	20	'Bremen'	2023-07-14
145	'Huber, Chr.'	21	'Augsburg'	2019-03-29
146	'Abele, I.'	22	'Senden'	2018-09-05
149	'Kircher, B.'	23	'Bochum'	2019-12-18
155	'Meier, W.'	24	'Stuttgart'	2023-04-02

- Das Ergebnis ist eine Relation mit ID, Name, Alter, Wohnort und Registrierungsdatum aller Mitglieder, die jünger als 25 Jahre sind.

Hinweis



$\sigma_{\text{Age} < 25}(\text{Member})$ entspricht in RelaX

`sigma Age < 25 (Member)`

b) Bestimmen Sie Name, Alter und Wohnort aller Mitglieder, d. h. Member, die 25 Jahre oder älter sind und nicht in Innsbruck wohnen. Geben Sie dazu

- den entsprechenden Relationenalgebra-Ausdruck und
- sein von Ihnen zunächst ohne Rückgriff auf RelaX berechnetes Ergebnis an, das Sie anschließend mithilfe von RelaX überprüfen.

Lösung



i. $\pi_{\text{Name, Age, Place}}(\sigma_{\text{Age} > 25 \wedge \text{Place} \neq \text{'Innsbruck'}}(\text{Member}))$

ii. Member.Name	Member.Age	Member.Place
'Schulze, B.'	26	'Stuttgart'
'Mons, F.'	26	'Essen'
'Meier, K.'	27	'Heidelberg'
'Karstens, L.'	27	'Hamburg'

Hinweis

$\pi_{\text{Name, Age, Place}}(\sigma_{\text{Age} > 25 \wedge \text{Place} \neq \text{'Innsbruck'}}(\text{Member}))$ entspricht in Relax

`pi Name, Age, Place`

`(sigma Age > 25 AND Place != 'Innsbruck' (Member))`

c) Bestimmen Sie

$\pi_{\text{Name, RegistrationDate}}(\sigma_{2023-01-01 \leq \text{RegistrationDate} \wedge \text{RegistrationDate} \leq 2023-12-31}(\text{Member}))$.

Geben Sie dazu

- das von Ihnen zunächst ohne Rückgriff auf Relax berechnete Ergebnis, das Sie anschließend mithilfe von Relax überprüfen, und
- eine Beschreibung von diesem in eigenen Worten an.

Hinweis

Folgen Sie bitte Ihrer Intuition, dass wir zwei Daten D_1 und D_2 im Format YYYY-MM-DD miteinander vergleichen können. Damit können wir die uns bekannte mathematische Notation nutzen, um mithilfe der Vergleichszeichen $<$, \leq , $=$, \geq und $>$ zu bestimmen, ob D_1 kleiner, größer oder gleich D_2 ist, d. h. ob D_1 zeitlich vor oder nach D_2 liegt bzw. gleich D_2 ist.

Hinweis

Nutzen Sie die Relax-Funktion *date*, um ein als *String* angegebenes Datum im Format YYYY-MM-DD in ein *Datum im Format YYYY-MM-DD* umzuwandeln.

Lösung

i. Member.Name	Member.RegistrationDate
'Schmidt, M.'	2023-07-14
'Meier, W.'	2023-04-02
'Gerstner, M.'	2023-05-30

- Das Ergebnis ist eine Relation mit Name und Registrierungsdatum aller Mitglieder, die in diesem Jahr Mitglied geworden sind.

Hinweis



$\pi_{\text{Name, RegistrationDate}} (\sigma_{2023-01-01 \leq \text{RegistrationDate} \wedge \text{RegistrationDate} \leq 2023-12-31} (\text{Member}))$

entspricht in Relax

```
pi Name, RegistrationDate
(sigma RegistrationDate >= date('2023-01-01') AND
RegistrationDate <= date('2023-12-31') (Member))
```

- d) Bestimmen Sie alle Orte, in denen zumindest ein Mitglied wohnt und es zumindest einen Ortsverband, d. h. Chapter, gibt. Geben Sie dazu
- den entsprechenden Relationenalgebra-Ausdruck und
 - sein von Ihnen zunächst ohne Rückgriff auf Relax berechnetes Ergebnis an, das Sie anschließend mithilfe von Relax überprüfen.

Lösung



i. $\pi_{\text{Place}}(\text{Member}) \cap \pi_{\text{Place} \leftarrow \text{Location}}(\text{Chapter})$

ii. Chapter.Place
'Augsburg'
'Innsbruck'
'Heidelberg'

Hinweis



$\pi_{\text{Place}}(\text{Member}) \cap \pi_{\text{Place} \leftarrow \text{Location}}(\text{Chapter})$ entspricht in Relax

```
pi Place (Member) intersect pi Place<-Location (Chapter)
```

- b) ☐ ★★ Was bedeutet, dass die Operanden der Mengenoperationen Vereinigung (\cup), Differenz ($-$) sowie Durchschnitt (\cap) strukturgleich sein müssen?

Lösung



Um die Mengenoperationen Vereinigung (\cup), Differenz ($-$) sowie Durchschnitt (\cap) auf den Relationen R und S durchführen zu können, müssen beide miteinander kompatibel sein. Dies ist genau dann der Fall, wenn

- beide Relationenschemata gleiche viele Attribute haben und
- die Domäne dieser Attribute paarweise identisch sind.

Seien als Beispiel $R(r_1, r_2, \dots, r_n)$ und $S(s_1, s_2, \dots, s_m)$ Relationen. Dann sind R und S strukturgleich, wenn

- $n = m$ und

2. die Domänen

$$\text{dom}(s_1), \text{dom}(s_2), \dots, \text{dom}(s_n), \text{dom}(r_1), \text{dom}(r_2), \dots, \text{dom}(r_m)$$

jeweils paarweise identisch sind, d. h.

$$\text{dom}(r_1) = \text{dom}(s_1), \text{dom}(r_2) = \text{dom}(s_2), \dots, \text{dom}(r_n) = \text{dom}(s_m).$$

Wie in Unteraufgabe a) d) genutzt, können wir Strukturgleichheit teilweise mithilfe einer Projektion und gegebenenfalls einer (impliziten) Attribut-Umbenennung erhalten.

Hausaufgabenteil (Zuhause zu lösen; Abgabe nötig)

Hinweis



Greifen Sie gerne auf das webbasierte Übungstool für relationale Algebra *RelaX*^a zurück, um auf den gegebenen Relationen Relationenalgebra-Operationen auszuführen und den zu diesen zugehörigen Operatorbaum zu generieren.

^a<https://dbis-uibk.github.io/relax>

Aufgabe 1 (Music Streaming Service)

[6 Punkte]

Gegeben sei ein Relationenmodell mit folgenden Relationenschemata:

Genre(GenreId, Name)
Artist(ArtistId, Name)
Album(AlbumId, Title, ArtistId)
Track(TrackId, Name, AlbumId, GenreId, Miliseconds, Bytes, UnitPrice)
Customer(CustomerId, FirstName, LastName, Address, Email)
Invoice(InvoiceId, CustomerId, InvoiceDate, Total)
InvoiceParts(InvoicePartId, InvoiceId, TrackId, UnitPrice, Quantity)
Playlist(PlaylistId, Name)
PlaylistContent(PlaylistId, TrackId)

Hinweis



Unter

dbis-uibk.github.io/relax/calc/gist/97d011026e4e35ce512d86d9c6b8a0c3

können Sie die Datenbank „Music Streaming Service“^a in RelaX öffnen, um auf ihren Relationen Relationenalgebra-Operationen auszuführen.

^ahttps://gist.github.com/antstei/97d011026e4e35ce512d86d9c6b8a0c3#file-music_streaming_service_description-md

Geben Sie für jede der folgenden Unteraufgaben

1. den entsprechenden Relationenalgebra-Ausdruck,
2. sein mithilfe von RelaX berechnetes Ergebnis sowie
3. die Anzahl der Tupel der Resultsrelation an.

Hinweis



Listen Sie bitte lediglich 10 exemplarische Tupel der Resultsrelation auf, sollte das Ergebnis eines Relationenalgebra-Ausdrucks mehr als 15 Tupel umfassen.

Hinweis



Geben Sie bitte sämtliche Operatoren in ihrer ausgeschriebenen Notation an, d. h. beispielsweise π anstatt π und join anstatt \Join .

- a) **0.5 Punkte** Bestimmen Sie alle Rechnungen, d. h. Invoice, deren Gesamtsumme, d. h. Total, kleiner als 5 Euro ist. Geben Sie dazu entsprechendes Ergebnis mit Relationenschema

R(Invoice.InvoiceId, Invoice.CustomerId, Invoice.InvoiceDate, Invoice.Total)
an.

Abgabe



1a_query.txt

1a_result.txt

Lösung



```
pi InvoiceId, CustomerId, InvoiceDate, Total (sigma Total < 5 (Invoice))
```

Invoice.InvoiceId	Invoice.CustomerId	Invoice.InvoiceDate	Invoice.Total
69	25	2009-10-25	0.99
70	26	2009-11-07	1.98
71	28	2009-11-07	1.98
72	30	2009-11-08	3.96
90	21	2010-01-26	0.99
91	22	2010-02-08	1.98
92	24	2010-02-08	1.98
93	26	2010-02-09	3.96
198	6	2011-05-20	3.96

9 Tupel gesamt

- b) **0.5 Punkte** Bestimmen Sie alle Rechnungen, die mit November 2009 ausgestellt wurden. Geben Sie dazu entsprechendes Ergebnis mit Relationenschema

R(Invoice.InvoiceId, Invoice.InvoiceDate, Invoice.Total, Customer.LastName)
an.

Abgabe



1b_query.txt

1b_result.txt

Lösung



```
pi InvoiceId, InvoiceDate, Total, LastName
((sigma InvoiceDate >= date('2009-11-01') and
  InvoiceDate <= date('2009-11-30') (Invoice))
join Invoice.CustomerId = Customer.CustomerId Customer)
```

Invoice.InvoiceId	Invoice.InvoiceDate	Invoice.Total	Customer.LastName
70	2009-11-07	1.98	'Cunningham'
71	2009-11-07	1.98	'Barnett'
72	2009-11-08	3.96	'Francis'

74	2009-11-12	8.91	'Lefebvre'
4 Tupel gesamt			

- c) 0.5 Punkte Bestimmen Sie alle Titel, d. h. Tracks, des Genres „Rock“, die auch tatsächlich gekauft wurden. Geben Sie dazu entsprechendes Ergebnis mit Relationenschema

R(Track.Name, Track.TrackId)

an.

Abgabe



1c_query.txt

1c_result.txt

Lösung



```

pi Track.Name, Track.TrackId
((Track join Track.TrackId = InvoiceParts.TrackId InvoiceParts)
 join Track.GenreId = Genre.GenreId (sigma Genre.Name = 'Rock' (Genre)))

```

Track.Name	Track.TrackId
'In Your Honor'	989
'No Way Back'	990
'The Last Song'	994
'Free Me'	995
'Still'	999
'What If I Do?'	1000
'Friend Of A Friend'	1003
'Over And Out'	1004
'Virginia Moon'	1006
'Razor'	1008
...	

64 Tupel gesamt

- d) 0.5 Punkte Bestimmen Sie für jede Playlist, welche Titel sie enthält. Geben Sie dazu entsprechendes Ergebnis mit Relationenschema

R(Playlist.Name, Track.Name)

an.

Abgabe



1d_query.txt

1d_result.txt

Lösung



```
pi Playlist.Name, Track.Name
(Playlist join Playlist.PlaylistId = PlaylistContent.PlaylistId
  PlaylistContent
  join Track.TrackId = PlaylistContent.TrackId Track)

Playlist.Name  Track.Name
'Music'        'In Your Honor'
'Music'        'No Way Back'
'Music'        'Best Of You'
'Music'        'DOA'
'Music'        'Hell'
'Music'        'The Last Song'
'Music'        'Free Me'
'Music'        'Resolve'
'Music'        'The Deepest Blues Are Black'
'Music'        'End Over End'
...

372 Tupel gesamt
```

- e) **1 Punkt** Bestimmen Sie alle von Kund*innen, d. h. Customer, deren Nachname mit „A“ oder „B“ beginnt, gekauften Lieder. Geben Sie dazu entsprechendes Ergebnis mit Relationenschema

R(Customer.LastName, Track.Name, Artist.Name, Album.Title)

an.

Abgabe



1e_query.txt

1e_result.txt

Lösung



```
pi LastName, Track.Name, Artist.Name, Album.Title
(sigma LastName < 'C' Customer join Customer.CustomerId = Invoice.CustomerId
  Invoice
  join Invoice.InvoiceId = InvoiceParts.InvoiceId InvoiceParts
  join InvoiceParts.TrackId = Track.TrackId Track
  join Track.AlbumId = Album.AlbumId Album
  join Album.ArtistId = Artist.ArtistId Artist)

Customer.LastName  Track.Name      Artist.Name      Album.Title
'Barnett'          'So Central Rain' 'R.E.M.'         'The Best Of R.E.M.: The IRS Years'
'Barnett'          'Pretty Persuasion' 'R.E.M.'         'The Best Of R.E.M.: The IRS Years'
'Barnett'          'Gimmie Shelters' 'Rolling Stones' 'No Security'
'Barnett'          'Thief In The Night' 'Rolling Stones' 'No Security'
'Barnett'          'Out Of Tears' 'Rolling Stones' 'Voodoo Lounge'
'Barnett'          'In Your Honor' 'Foo Fighters' 'In Your Honor [Disc1]'
'Barnett'          'Free Me' 'Foo Fighters' 'In Your Honor [Disc1]'
```

'Brown'	'Californication'	'Red Hot Chili Peppers'	'Californication'
'Brown'	'Road Trippin'	'Red Hot Chili Peppers'	'Californication'
'Bernard'	'Don't Go Back'	'R.E.M.'	'The Best Of R.E.M.: The IRS Years'
'Bernard'	'I Believe'	'R.E.M.'	'The Best Of R.E.M.: The IRS Years'

11 Tupel gesamt

- f) **1 Punkt** Bestimmen Sie alle Lieder, die nicht in der Playlist mit der PlaylistId 5 enthalten sind, mit zwei verschiedenen Relationenalgebra-Ausdrücken, indem Sie

- 1) bei erster Variante einen Semi-Join und
- 2) bei zweiter Variante einen Outer-Join verwenden.

Geben Sie dazu entsprechende Ergebnisse mit Relationenschema

$R(\text{Track.TrackId}, \text{Track.Name}, \text{Track.UnitPrice})$

an.

Abgabe



1f_query_1.txt

1f_query_2.txt

1f_result.txt

Lösung



- 1) erster Variante mit einem Semi-Join

```
pi TrackId, Name, UnitPrice
  (Track - (Track left semi join (sigma PlaylistId = 5 PlaylistContent)))
```

- 2) zweite Variante mit einem Outer-Join

```
pi TrackId, Name, UnitPrice
  (sigma PlaylistId = null
    (Track left outer join (sigma PlaylistId = 5 (PlaylistContent))))
```

Track.TrackId	Track.Name	Track.UnitPrice
989	'In Your Honor'	0.99
990	'No Way Back'	0.99
991	'Best Of You'	0.99
992	'DOA'	0.99
993	'Hell'	0.99
994	'The Last Song'	0.99
995	'Free Me'	0.99
996	'Resolve'	0.99
997	'The Deepest Blues Are Black'	0.99
998	'End Over End'	0.99
...		

62 Tupel gesamt

- g) 1 Punkt Bestimmen Sie alle Künstler*innen, d. h. Artist, deren Titel
- durchschnittlich über 4 Minuten und 10 Sekunden lang sind *und*
 - durchschnittliche eine Dateigröße unter 8,5 MB haben.

Geben Sie dazu entsprechendes Ergebnis mit Relationenschema

R(Artist.ArtistId, Artist.Name)

an.

Abgabe



1g_query.txt

1g_result.txt

Lösung



```
pi ArtistId, Name
(sigma avg_duration > 250000
  (gamma Artist.ArtistId, Artist.Name; avg(Milliseconds) -> avg_duration
    (Track natural join Album join Album.ArtistId = Artist.ArtistId Artist))
  )

intersect

pi ArtistId, Name
(sigma avg_size < 8500000
  (gamma Artist.ArtistId, Artist.Name; avg(Bytes) -> avg_size
    (Track natural join Album join Album.ArtistId = Artist.ArtistId Artist))
  )

Artist.ArtistId  Artist.Name
84               'Foo Fighters'

1 Tupel gesamt
```

- h) 1 Punkt Bestimmen Sie alle Kund*innen, die nach dem 1. Januar 2010 mindestens drei Titel eines Albums gekauft haben.

Geben Sie dazu entsprechendes Ergebnis mit Relationenschema

R(Customer.CustomerId, Customer.LastName, Track.AlbumId, Album.Name, Quantity)

an, wobei Quantity der Anzahl der gekauften Titel und Album.Name dem umbenannten Attribut Album.Title entspricht.

Abgabe



1h_query.txt

1h_result.txt

Lösung



```
rho Name <- Title
(sigma Quantity > 2
  (gamma CustomerId, LastName, AlbumId, Title; count(AlbumId) -> Quantity
    (Customer natural join (sigma InvoiceDate > date('2010-01-01') Invoice)
      natural join InvoiceParts
      natural join Track
      natural join Album)))
```

Customer.CustomerId	Customer.LastName	Track.AlbumId	Album.Name	Quantity
10	'Martins'	238	'The Best Of 1980-1990'	3
14	'Philips'	190	'The Best Of R.E.M.: The IRS Years'	4
20	'Mille'	194	'By The Way'	3
26	'Cunningham'	239	'War'	3

4 Tupel gesamt

Aufgabe 2 (Optimierung von Ausdrücken)

[4 Punkte]

Gegeben sei dasselbe Relationenmodell wie in Aufgabe 1 mit folgenden Relationenschemata:

```
Genre(GenreId, Name)
Artist(ArtistId, Name)
Album(AlbumId, Title, ArtistId)
Track(TrackId, Name, AlbumId, GenreId, Miliseconds, Bytes, UnitPrice)
Customer(CustomerId, FirstName, LastName, Address, Email)
Invoice(InvoiceId, CustomerId, InvoiceDate, Total)
InvoiceParts(InvoicePartId, InvoiceId, TrackId, UnitPrice, Quantity)
Playlist(PlaylistId, Name)
PlaylistContent(PlaylistId, TrackId)
```

Hinweis



Unter

dbis-uibk.github.io/relax/calc/gist/97d011026e4e35ce512d86d9c6b8a0c3

können Sie nach wie vor die Datenbank „Music Streaming Service“^a in RelaX öffnen, um auf ihren Relationen Relationenalgebra-Operationen auszuführen, unter

dbis-uibk.github.io/relax/calc/gist/bc60c641f4006967df49713cd5c25a72

die entsprechenden Relationenschemata, d. h. die leere, nicht mit Tupeln befüllte Datenbank „Music Streaming Service“.

^ahttps://gist.github.com/antstei/97d011026e4e35ce512d86d9c6b8a0c3#file-music_streaming_service_description-md

Ziel dieser Aufgabe ist es, mit drei verschiedenen effizienten Relationenalgebra-Ausdrücken

alle Kund*innen, d. h. Customer, zu bestimmen, die nach dem 01.01.2010 mindestens einen Titel, d. h. Track, des Genres „Rock“ gekauft haben,

entsprechendes Ergebnis mit Relationenschema

$R(\text{Customer.FirstName}, \text{Customer.LastName}, \text{Track.Name}, \text{Invoice.InvoiceDate})$

anzugeben und den zugehörigen Operatorbaum zu analysieren.

Hinweis



Testen Sie Ihre ineffizienten Relationenalgebra-Ausdrücke auf der leeren Datenbank, da diese sehr ressourcenintensiv sind.

- a) 0.5 Punkte Um die Effizienz eines Relationenalgebra-Ausdrucks abschätzen können, ist es notwendig, dass wir die Anzahl der Tupel seiner Operanden, d. h. Relationen, kennen. Bestimmen Sie aus diesem Grund mithilfe eines passenden Relationenalgebra-Ausdrucks die Anzahl der Tupel der folgenden Relationen in der mit Daten befüllten Datenbank „Music Streaming Service“.

Ergänzen Sie dazu in der Tabelle `2a_table.txt`^{OLAT} die fehlenden Werte.

Abgabe



2a_table.txt

Lösung



Die Anzahl der Tupel einer Relation können wir bestimmen, indem wir die Aggregatfunktion count ohne Gruppierung auf dieser ausführen, d. h. $\text{gamma} ; \text{count}(A) \rightarrow c(R)$, wobei A einem beliebigen Attribut der Relation R entspricht.

Relation	Anzahl Tupel
Artist	5
Album	11
Track	147
Rock Tracks	114
Playlist	3
PlaylistContent	379
Genre	9
Invoice	32
Invoice after 2010-01-01	25
InvoiceParts	99
Customer	27

- b) 1 Punkt Formulieren Sie Ihren ersten, ineffizienten Relationenalgebra-Ausdruck, um alle Kund*innen zu bestimmen, die nach dem 01.01.2010 mindestens einen Titel des Genres „Rock“ gekauft haben,

und entsprechendes Ergebnis mit Relationenschema

`R(Customer.FirstName, Customer.LastName, Track.Name, Invoice.InvoiceDate)`

anzugeben.

Greifen Sie dazu auf

- 1) *genau eine* Projektion,
- 2) *genau eine* Selektion,
- 3) und beliebig viele Kreuzprodukte

zurück, sodass Ihr Relationenalgebra-Ausdruck der Form


$$\pi.(\sigma.(A \times \dots \times Z))$$


entspricht, wobei A, \dots, Z Relationen sind.

Generieren Sie anschließend den zu Ihrem Relationenalgebra-Ausdruck zugehörigen Operatorbaum und bestimmen Sie für jeden seiner Knoten die Anzahl der Tupel der jeweiligen (Teil-)Resultatsrelation.

Abgabe



 2b_query.txt

 2b_tree.pdf

Lösung

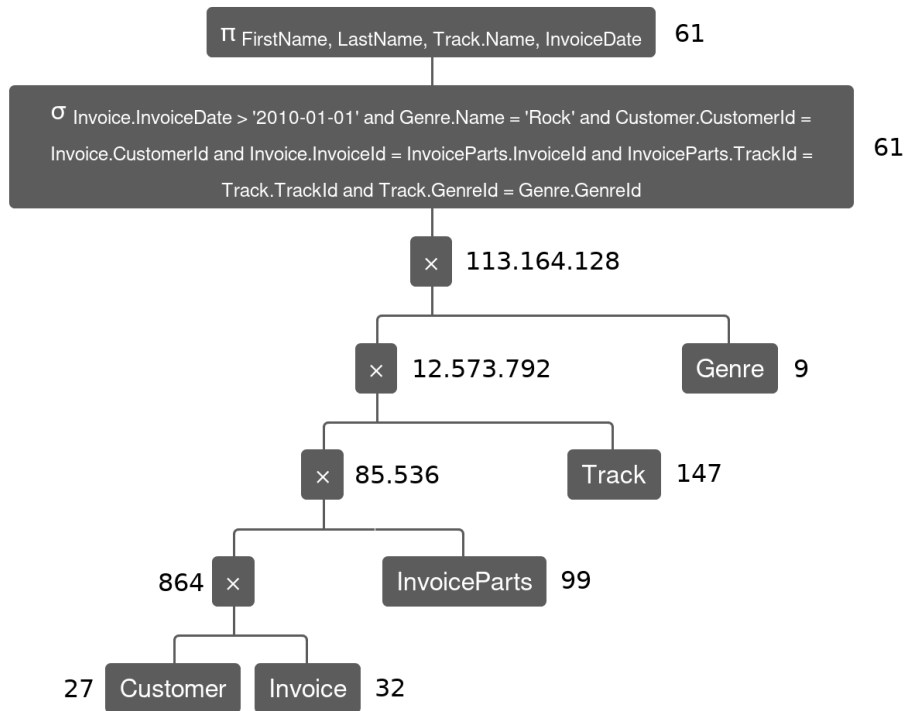


`pi FirstName, LastName, Track.Name, InvoiceDate`

```

(sigma Invoice.InvoiceDate > date('2010-01-01') and
  Genre.Name = 'Rock' and
  Customer.CustomerId = Invoice.CustomerId and
  Invoice.InvoiceId = InvoiceParts.InvoiceId and
  InvoiceParts.TrackId = Track.TrackId and
  Track.GenreId = Genre.GenreId)
  (Customer x Invoice x InvoiceParts x Track x Genre))

```



- c) **1 Punkt** Überarbeiten Sie Ihren in der vorherigen Unteraufgabe formulierten ineffizienten Relationalalgebra-Ausdruck, sodass dieser die Selektionen – die den jeweiligen Joins entsprechen – direkt nach dem Kreuzprodukt ausführt.

Greifen Sie dazu auf noch *keinen expliziten* Join zurück, sondern auf sein semantisches Äquivalent

$$A \bowtie_c B = \sigma_c(A \times B).$$

Generieren Sie anschließend den zu Ihrem überarbeiteten, effizienteren Relationalalgebra-Ausdruck zugehörigen Operatorbaum und bestimmen Sie für jeden seiner Knoten die Anzahl der Tupel der jeweiligen (Teil-)Resultatsrelation.

Abgabe



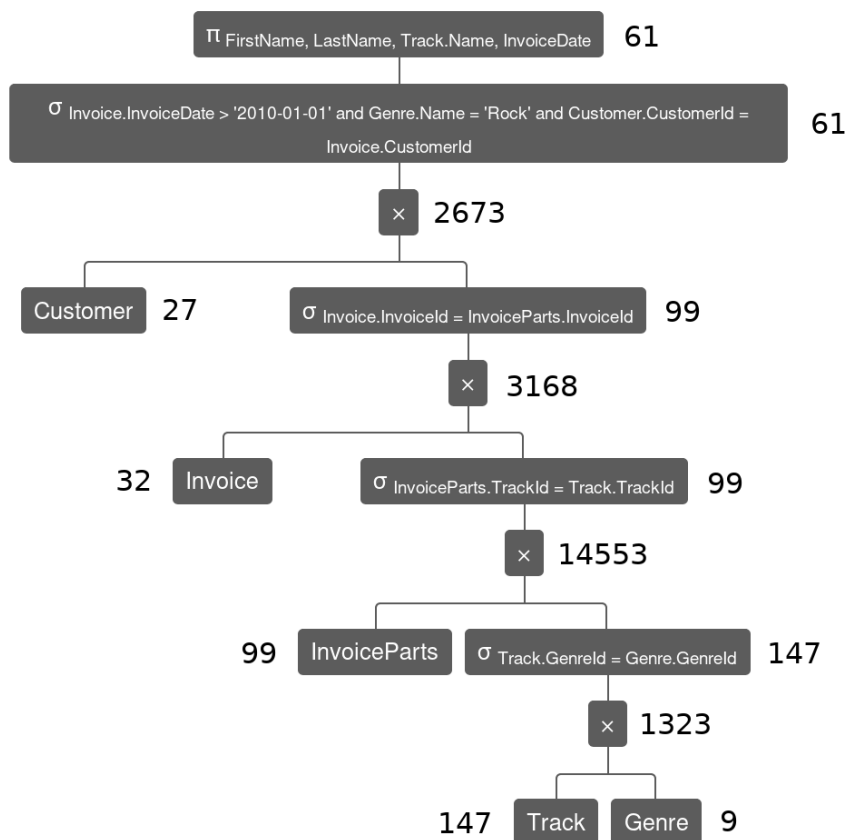
2c_query.txt

2c_tree.pdf

```

pi FirstName, LastName, Track.Name, InvoiceDate
  (sigma Invoice.InvoiceDate > date('2010-01-01') and
    Genre.Name = 'Rock' and
    Customer.CustomerId = Invoice.CustomerId
    (Customer x
      (sigma Invoice.InvoiceId = InvoiceParts.InvoiceId
        (Invoice x
          (sigma InvoiceParts.TrackId = Track.TrackId
            (InvoiceParts x
              (sigma Track.GenreId = Genre.GenreId
                (Track x Genre))))))))))

```



d) 1.5 Punkte Wie Sie in der vorigen Unteraufgabe bemerkt haben, lohnt es sich, so früh wie möglich die Ergebnismenge durch Selektion zu verringern. Überarbeiten Sie aus diesem Grund Ihren bereits in der vorherigen Unteraufgabe optimierten Relationenalgebra-Ausdruck, indem Sie




- 1) anstatt von Kreuzprodukten in Verbindung mit einer Selektion auf Joins zurückgreifen,
- 2) das Datum und Genre so früh wie möglich selektieren und
- 3) die Join-Reihenfolge optimieren.

Geben Sie für jeden dieser Punkte an, warum diese die Abfrage optimieren. Generieren Sie

anschließend den zu Ihrem erneut überarbeiteten, effizienten Relationenalgebra-Ausdruck zugehörigen Operatorbaum und bestimmen Sie für jeden seiner Knoten die Anzahl der Tupel der jeweiligen (Teil-)Resultatsrelation.

Abgabe



 2d_query.txt
 2d_tree.pdf
 2d_explanation.txt

Hinweis

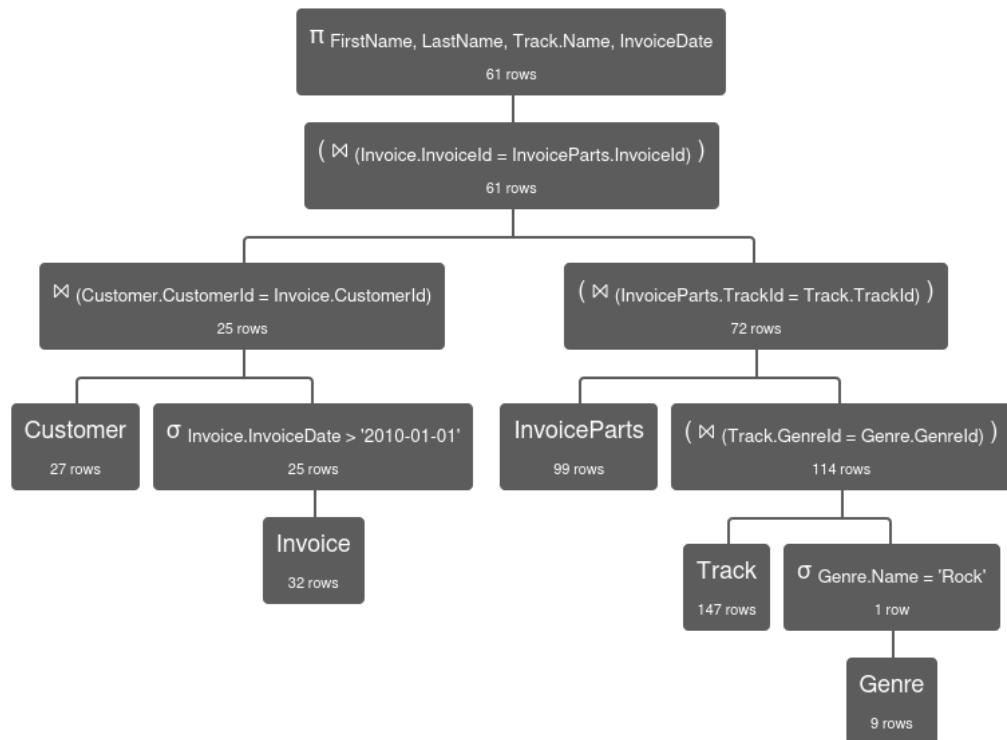


Versuchen Sie Ihren in der ersten Unteraufgabe formulierten ineffizienten Relationenalgebra-Ausdruck sowie den von Ihnen in dieser Unteraufgabe optimierte auf der mit Daten befüllten Datenbank „Music Streaming Service“ auszuführen. Bemerken Sie den Unterschied?

Lösung



```
pi FirstName, LastName, Track.Name, InvoiceDate
  (Customer join (Customer.CustomerId = Invoice.CustomerId)
    (sigma Invoice.InvoiceDate > date('2010-01-01') Invoice)
    join (Invoice.InvoiceId = InvoiceParts.InvoiceId)
      (InvoiceParts join (InvoiceParts.TrackId = Track.TrackId)
        (Track join (Track.GenreId = Genre.GenreId) (
          sigma Genre.Name = 'Rock' Genre))))
```



Warum stellen die Änderungen Optimierungen dar?

- 1) Der Join-Operator kann vorhandene Indexstrukturen verwenden bzw. gegebenenfalls sogar „spontan“ für die jeweilige Operation temporäre Indizes erstellen, wenn dies rentabel ist.
- 2) Die Selektionen so früh wie möglich durchzuführen, reduziert die Anzahl der Tupel in den Zwischenergebnissen.
- 3) Die Join-Reihenfolge ist wichtig, jedoch komplex zu bestimmen. Ziel ist hier auch, die Zwischenergebnisse kleinzuhalten, jedoch kann das DBMS oft nur (über Heuristiken) abschätzen, wie groß die Daten nach dem Join sein werden, z. B. wie viele Tupel entsprechen der Joinbedingung.

Wichtig: Laden Sie bitte Ihre Lösung in OLAT hoch und geben Sie mittels der Ankreuzliste auch unbedingt an, welche Aufgaben Sie gelöst haben. Die Deadline dafür läuft am Vortag des Proseminars um 16:00 ab.