

All RA signs:

|  |  |
| --- | --- |
| Projection | Π |
| Where clause | σ |
| rename relation \ rename column | ρ |
| rename column operator | ← |
| Order by | τ |
| group by | γ |
| AND | ∧ |
| OR | ∨ |
| Not | ∨ |
| OR | ¬ |
| equal | = |
| Not equal | ≠ |
| Bigger or equal | ≥ |
| less or equal | less or equal |
| Cross join | ⨯ |
| Natural join | ⨝ |
| left join | ⟕ |
| Right join | ⟖ |
| Full outer join | ⟗ |
| Left semi join | ⋉ |
| Right semi join | ⋊ |

Relation algebra:

We can use actual online database that have those tables

In this we site there is an online calculator that we can use exiting database of create our own.

The database is called group.

[http://dbis-uibk.github.io/relax/calc.htm#](http://dbis-uibk.github.io/relax/calc.htm)

This is the database and the data I used for question 1.

In question 1.A we are asked to use client and file relations.

-- this is the database from maman 12 ( 2016c)

group: bank example

description[[ the data for this dataset was generated using <http://www.generatedata.com/>

\* the relation \_Customers\_ contains basic information about the customers of the bank.

\* the relation \_Accounts\_ contains the basic information of a single account. Note that a customer can have any number of accounts.

\* the relation \_PremiumCustomers\_ contains the customer-ids of all customers with a total balance over 1000

]]

lawyer = { lname:string sdate:date hbilling:number partner:date

ahoron 2016-01-01 200 2015-01-01

moshes 2016-01-01 200 null

yontan 2016-01-01 200 null

amir 2016-02-01 200 2015-01-01

amit 2016-03-01 200 2015-01-01

}

client = { cname:string tel:string address:string

eli 0504831232 rehovot

lital 0504831231 rehovot

mor 0504831234 rehovot

danny 0504831235 rehovot

}

File = { fid:number cname:string description:string status:date lname:string

103212 eli stolen null amir

103211 lital barelegry null amir

103213 danny kidnap null amir

103214 mor stolen null amir

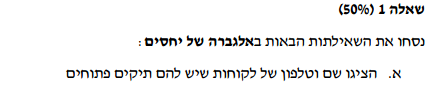
103215 danny stolen 2016-06-02 amir

}

* Make sure that the table column name are correct , like cname

Otherwise the natural join will not work

The way I solved the relational algebra is to first do a sql query and then converts it to RA.



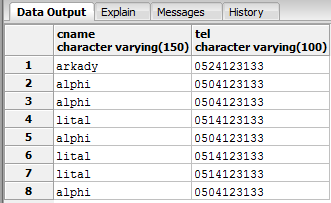
select client.cname, client.tel from file

natural join client

where file.status is null

Relational algebra does not have inner join, on ..

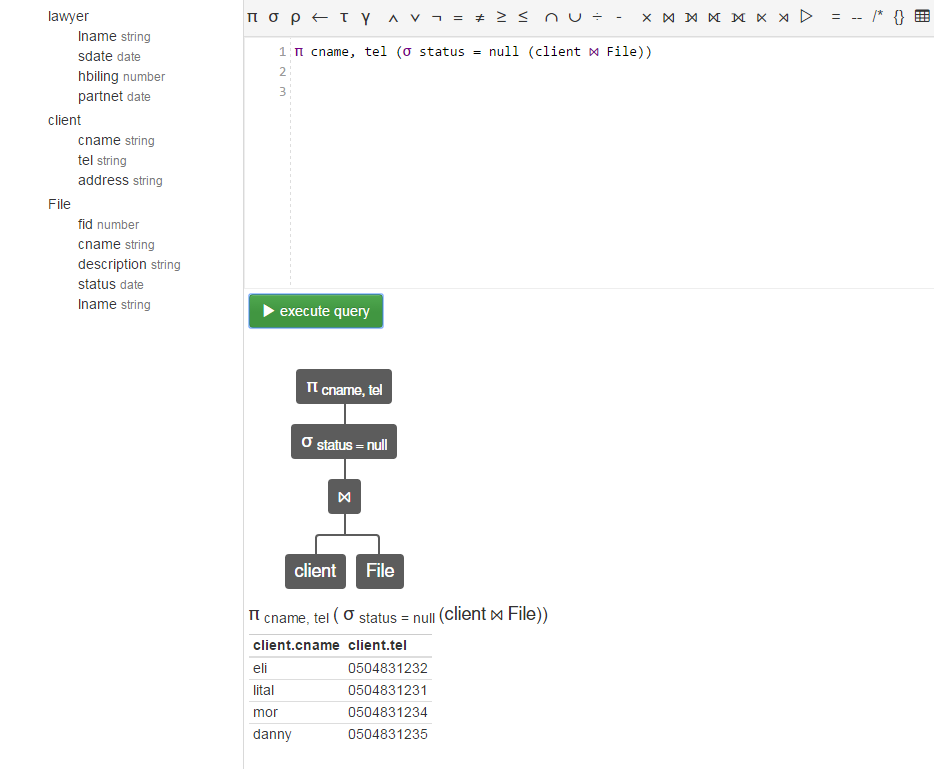
It does have natural join, which is inner join on common attribute names.



Now we can construct our RA:

π cname, tel (σ status = null (client ⨝ file))

This is how it looks in the calculator

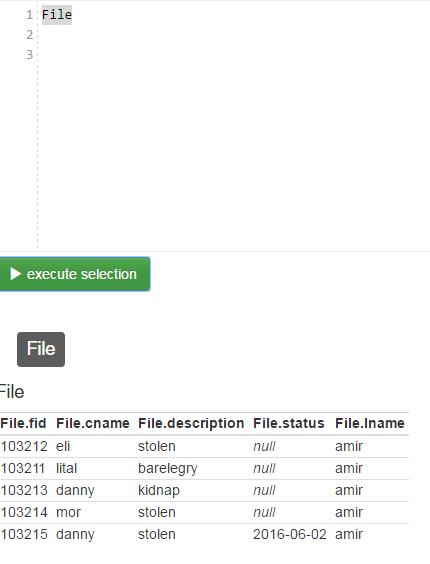


On the left we see our relations in the group.

And we can execute our RA query

Btw , to view the content of a relation just write the name of it and execute

For example:





The sql query for this question is:

select file.fid, file.lname from file

left join lonfile

on file.fid = lonfile.fid

where lonfile.lname is null

This query uses left join , but It appears that the book does not show that we can use left and right join

The other solution is to use projection with substraction:

In SQL it is like writing the following query:

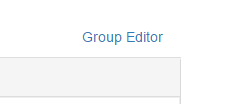
select fid, lname from file where

not exists ( select fid, lname from lonfile where file.fid = lonfile.fid )

We can see also an example for this query in the learning book at page 134

Example 6.12 for query number 4.

Back to our RA calculator on the web:



Press group editor, because we need to add some data to the lonfile relation, which we did not define yet.

I change two relations, File and added the lonfile:

File = { fid:number cname:string description:string status:date lname:string

103212 eli stolen null amir

103211 lital barelegry null moshes

103213 danny kidnap null amit

103214 mor stolen null yonatan

103215 danny stolen 2016-06-02 amir

}

lonfile = { fid:number lname:string responsibility:string

103212 ahoron second

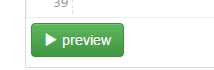
103211 amir second

103213 amit second

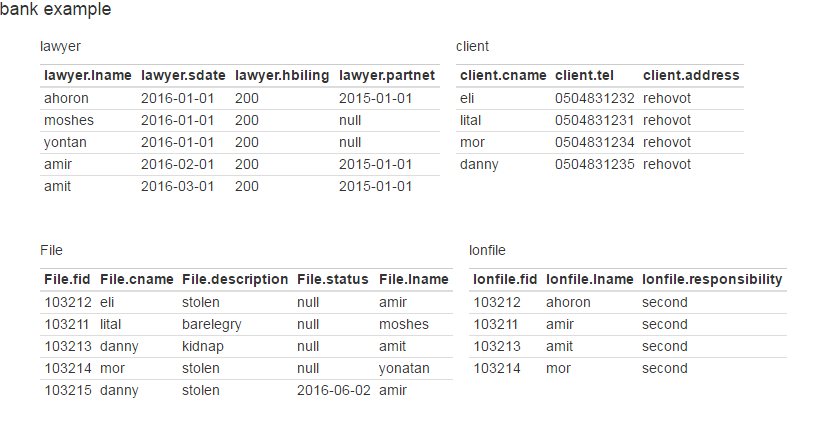
103214 mor second

}

Press preview:



Once you see the preview ok:

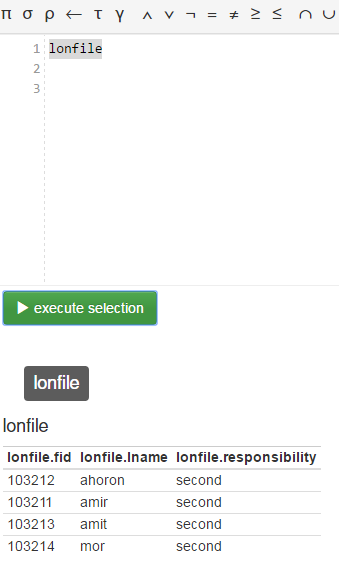


Press use in group editor:



And select at the top 

We can see the content of lonfile:



This query works fine:

π fid,lname (File) - π fid (lonfile)

but it is missing the lname.

We should do another query and make this query as sub query in sql

Like that :

select lname, fid from file

natural join (select fid, lname from file where

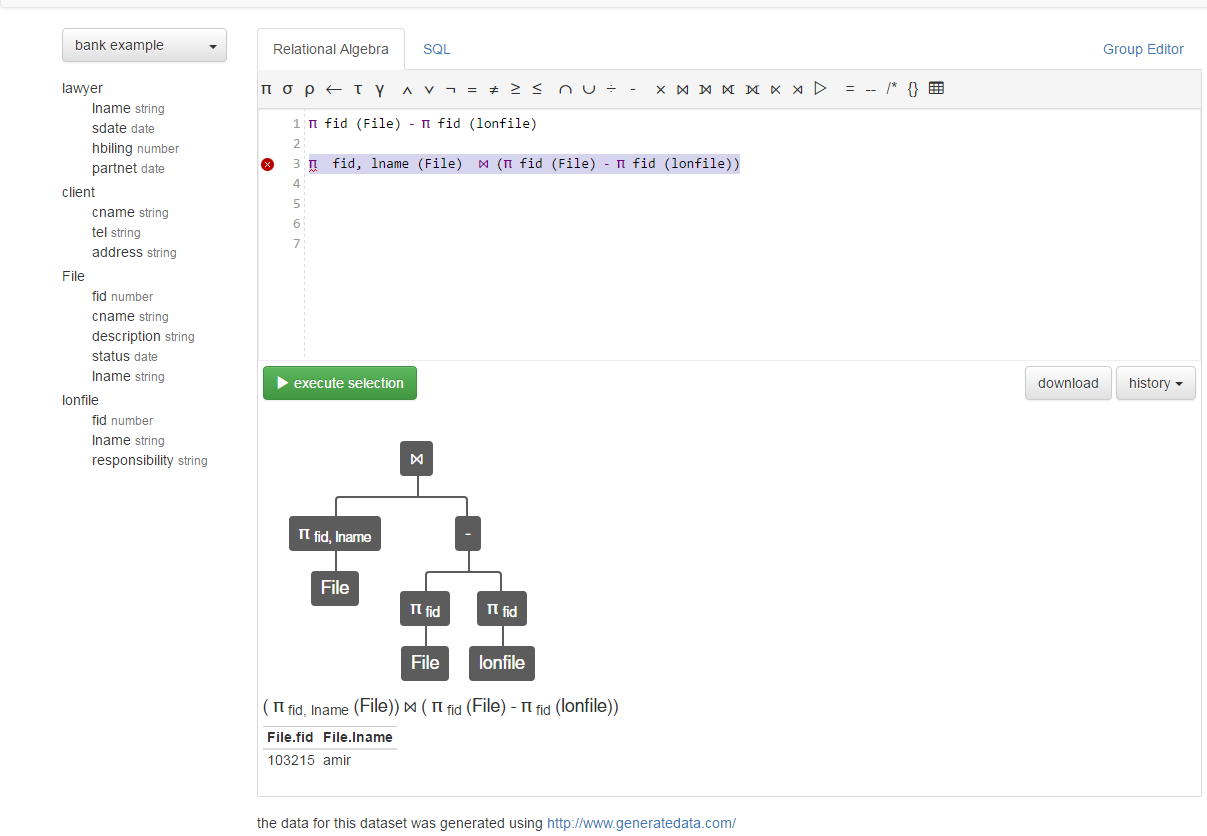
not exists ( select fid, lname from lonfile where file.fid = lonfile.fid )) as x

I used natural join to get more fields

In RA our result query will be:

π fid, lname (File) ⨝ (π fid (File) - π fid (lonfile))

This is how it looks in the calculator:





I will insert more lonfile sets:

INSERT INTO LonFile VALUES(120220, 'Eli', 'adviser');

INSERT INTO LonFile VALUES(113210, 'Eli', 'adviser');

INSERT INTO LonFile VALUES(914323, 'Eli', 'adviser');

INSERT INTO LonFile VALUES(431222, 'Eli', 'adviser');

INSERT INTO LonFile VALUES(312441, 'Eli', 'adviser');

INSERT INTO LonFile VALUES(112229, 'Eli', 'adviser');

INSERT INTO LonFile VALUES(114555, 'Eli', 'adviser');

INSERT INTO LonFile VALUES(114555, 'Eli', 'adviser');

INSERT INTO LonFile VALUES(120220, 'Eli', 'adviser');

INSERT INTO LonFile VALUES(610321, 'Eli', 'adviser');

INSERT INTO LonFile VALUES(120220, 'Eli', 'adviser');

INSERT INTO LonFile VALUES(412312, 'Eli', 'adviser');

INSERT INTO LonFile VALUES(120823, 'Eli', 'adviser');

INSERT INTO LonFile VALUES(810327, 'Eli', 'adviser');

INSERT INTO LonFile VALUES(810322, 'Eli', 'adviser');

INSERT INTO LonFile VALUES(610324, 'Eli', 'adviser');

INSERT INTO LonFile VALUES(810327, 'Eli', 'adviser');

We need to clean the data base first

delete from lonfile where lonfile.fid in (

select fid from file

natural join lonfile

where file.fid = lonfile.fid)

This sql delete from lonfile entries where the file lname is exist in the same fid in the lonfile fid.

Because It does not make sense that the file.lname in specific fid is exist as lonfile as lname in lonfile table.

Means, I assume that the above rows does not exists.

Current in the database I have 18 open cases



select count(\*) from file where status is null

‘Eli’ is working on 4 open cases as primary lawyer:

select count(\*) from file where lname = 'Eli' and status is null

and 12 case as secondary lawyer , which I now know that there are different cases.

Total we have 18 – ( 12+ 4) = 2.

Means , Eli is not working on all the cases yet ( we need 17 and 1)

Those are Eli’s open cases as a lawyer:

select \* from (

select fid from file where lname = 'Eli'

union

select fid from lonfile where lname = 'Eli' ) as xxx

order by fid

And those are all the cases:

|  |  |
| --- | --- |
|  |  |

So we are missing :

INSERT INTO LonFile VALUES(813212, 'Eli', 'adviser');

INSERT INTO LonFile VALUES(412121, 'Eli', 'adviser');

We can perform the delete again and see that the data is not deleted:

delete from lonfile where lonfile.fid in (

select fid from file

natural join lonfile

where file.fid = lonfile.fid)

select count(\*) from file where status is null

select count(\*) from file where lname = 'Eli' and status is null

select count(\*) from lonfile where lname = 'Eli'

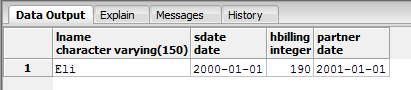
This returns now 18, 4 , 13 = 1

Means, Eli is a lawyer that handles all the open cases as any kind of responsibility:

My final SQL query is:

select \* from lawyer where

(select (select count(\*) as val1 from file where status is null) - (select count(\*) as val2 from file where lname = lawyer.lname and status is null) - (select count(\*) as val3 from lonfile where lname = lawyer.lname) as total\_count) = 1



Lets do the same for lawyer Ben

Checking the count:

select count(\*) from file where status is null

select count(\*) from file where lname = 'Ben' and status is null

select count(\*) from lonfile where lname = 'Ben'

Ben is missing 2 cases:

select \* from (

select fid from file where lname = 'Ben'

union

select fid from lonfile where lname = 'Ben' ) as xxx

order by fid

|  |  |
| --- | --- |
|  |  |

INSERT INTO LonFile VALUES(120333, 'Ben', 'adviser');

INSERT INTO LonFile VALUES(114555, 'Ben', 'adviser');

Checking that the data is ok by trying to delete:

INSERT INTO LonFile VALUES(114555, 'Ben', 'adviser');

delete from lonfile where lonfile.fid in (

select fid from file

natural join lonfile

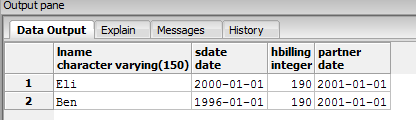
where file.fid = lonfile.fid)

Running our final query:

select \* from lawyer where

(select (select count(\*) as val1 from file where status is null) - (select count(\*) as val2 from file where lname = lawyer.lname and status is null) - (select count(\*) as val3 from lonfile where lname = lawyer.lname) as total\_count) = 1

Getting now two names:



Relational algebra:

We need to convert this query

select \* from lawyer where

(select (select count(\*) as val1 from file where status is null) - (select count(\*) as val2 from file where lname = lawyer.lname and status is null) - (select count(\*) as val3 from lonfile where lname = lawyer.lname) as total\_count) = 1

Into RA

**res = Gcount(\*) (σ status = null(File)) - Gcount(σ file.lname = lawyer.lname ∧ status = null(lawyer ⨝ File)) -**

**Gcount(\*) (σ lonfile.lname = lawyer.lname(lawyer ⨝ File))**

**σ (res = 1(Lawyer))**



In RA we don’t have group by( at least not in the learning book) , at least not in our learning book.

So we need to use here cross join, or Cartesian product

-- This is the first part of the query. which finds a lawyer which is not associate and earn more

-- for an hour then a lawyer which he is an associate.

select distinct l1.lname from lawyer as l1 , lawyer as l2

where l1.lname != l2.lname and l1.partner is null and l2.partner is not null and l1.hbilling > l2.hbilling

intersect

select cname from file

where status is null

-- this is the seconds part of the query , the above lowyer need to handle two active cases and he himself, a client in some kind

-- of case

select \* from client where cname = (select distinct l1.lname from lawyer as l1 , lawyer as l2

where l1.lname != l2.lname and l1.partner is null and l2.partner is not null and l1.hbilling > l2.hbilling

intersect

select cname from file

where status is null)

select count(lname) from file where status is null and lname = (select distinct l1.lname from lawyer as l1 , lawyer as l2

where l1.lname != l2.lname and l1.partner is null and l2.partner is not null and l1.hbilling > l2.hbilling

intersect

select cname from file

where status is null)

having count(lname) = 2

-- this is the final query - without CTE or functions - just copy paste from the above

select \* from client where cname = (select distinct l1.lname from lawyer as l1 , lawyer as l2

where l1.lname != l2.lname and l1.partner is null and l2.partner is not null and l1.hbilling > l2.hbilling

intersect

select cname from file

where status is null) and (select count(lname) from file where status is null and lname = (select distinct l1.lname from lawyer as l1 , lawyer as l2

where l1.lname != l2.lname and l1.partner is null and l2.partner is not null and l1.hbilling > l2.hbilling

intersect

select cname from file

where status is null)

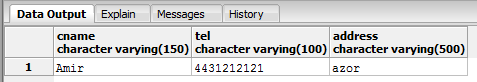
having count(lname) = 2) = 2

-- insert new data that will cause Amir to have two open cases:

INSERT INTO Client VALUES('david', '0504223110', 'Ramat-gan');

INSERT INTO File VALUES(120823, 'Stolen', 'Stolen bag from old lady' , NULL , 'david' , 'Amir');

Now, how do we check that the result of the above is a lawyer that in charge of 2 active cases?



In the RA group editor, I change the data to be:

lawyer = { lname:string sdate:date hbilling:number partner:date

ahoron 2016-01-01 200 2015-01-01

moshes 2016-01-01 200 null

yontan 2016-01-01 200 null

amir 2016-02-01 200 null

amit 2016-03-01 200 2015-01-01

Eli 2000-01-01 190 2001-01-01

}

client = { cname:string tel:string address:string

Eli 0504831232 rehovot

lital 0504831231 rehovot

mor 0504831234 rehovot

danny 0504831235 rehovot

amir 0504831235 rehovot

}

File = { fid:number cname:string description:string status:date lname:string

103212 Eli stolen null amir

103211 lital barelegry null amir

103213 danny kidnap null amir

103214 mor stolen null amir

103215 danny stolen 2016-06-02 amir

813212 Eli 'driving in circles' null amir

104211 lital barelegry null Eli

104211 amir barelegry null Eli

}

And this SQL query

select distinct l1.lname from lawyer as l1 , lawyer as l2

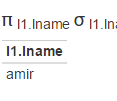
where l1.lname != l2.lname and l1.partner is null and l2.partner is not null and l1.hbilling > l2.hbilling

intersect

select cname from File

where status is null

Now returns:



In relational algebra , the first part is this:

π lname ( σ l1.lname ≠ l2.lname ∧ l1.partner = null and l2.partner ≠ null ∧ l1.hbilling > l2.hbilling (ρl1(lawyer) ⨯ ρl2(lawyer))

∩

π cname (σ status = null (File))

We can assign it using the 🡨 assign operator to a temporary name:

firstpart ← π lname ( σ l1.lname ≠ l2.lname ∧ l1.partner = null and l2.partner ≠ null ∧ l1.hbilling > l2.hbilling (ρl1(lawyer) ⨯ ρl2(lawyer))

∩

π cname (σ status = null (File))

Now we need to convert the

select count(lname) from file where status is null and lname

into RA:

Gcount(lname)(σ status = null ∧ lname = firstpart(File))

The result query will be:

Firstpart 🡨 π lname ( σ l1.lname ≠ l2.lname ∧ l1.partner = null and l2.partner ≠ null ∧ l1.hbilling > l2.hbilling (ρ l1 (lawyer) ⨯ ρ l2 (lawyer))

∩

π cname (σ status = null (File))

secondpart 🡨 Gcount(lname)(σ status = null ∧ lname = firstpart(File))

The result query will be:

π lname (σ cname = firstpart (File))

∧

secondpart = 2

Using the assignment operator I created two parts and used them.



I inserted more data

INSERT INTO Lawyer VALUES('Yaron', '12-1-2005', 100, '12-1-2010');

INSERT INTO Lawyer VALUES('Meir', '12-2-2015', 100, '12-1-2016');

INSERT INTO File VALUES(610321, 'Stolen', 'Stolen bag from old lady' , NULL , 'alphi' , 'Yaron');

INSERT INTO File VALUES(810322, 'Stolen', 'Stolen bag from old lady' , NULL , 'Eli' , 'Yaron');

INSERT INTO File VALUES(610324, 'Stolen', 'Stolen bag from old lady' , NULL , 'alphi' , 'Meir');

INSERT INTO File VALUES(810327, 'Stolen', 'Stolen bag from old lady' , NULL , 'Eli' , 'Meir')

The first query find pairs of lawyers than handle open

with yyy as

(

select lname from (

select lname, count(lname) from file

where status is null

group by lname

having count(lname) = 2) as ddd)

select lawyer.lname from lawyer,yyy where lawyer.lname = yyy.lname

and lawyer.partner is not null

the final SQL query is:

with xxx as

(

with yyy as

(

select lname from (

select lname, count(lname) from file

where status is null

group by lname

having count(lname) = 2) as ddd)

select lawyer.lname from lawyer

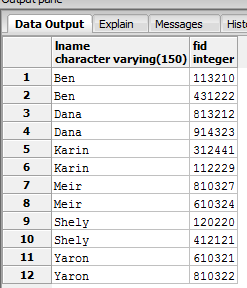
natural join yyy

where lawyer.lname = yyy.lname

and lawyer.partner is not null)

select file.lname, file.fid from file, xxx where file.lname = xxx.lname

order by xxx.lname



And in relational Algebra:

yyy = π lname, ρc(Gcount(lname)) (σ status = null ∧ c = 2)

xxx = π lname σ (l.lname = y.lname ∧ l.partner ≠ null)(ρl(Lawyer) ⨝ ρy(yyy)

π lname, fid (σ f.lname = x.lname) (ρf(File) ⨝ ρx(xxx))

