Name: KEY

Use the handout of three tables -- Vehicle, Own, Person -- to answer these.

0. **Assume that Vehicle only has the first two rows shown on the handout and that Person only has the first two rows shown on the handout. Show or otherwise describe the result on the natural join of Vehicle and Person.**

Natural join results in a pairing of rows in which a row of one table is paired with a row of another table if the two constituent rows are equal along all the same-named (and typed) attributes in the two tables. There are ZERO same-named attributes between Vehicle and Person, so all pairs of rows satisfy the constraint that ZERO same-named rows have the same value

Vehicle ⋈ Person							
VRN	Ma	Mo	Color	SSN	Name	Addr	Phone
123	Honda	Hawk	Red	abc	Dave	Birch	xxx
123	Honda	Hawk	Red	bcd	Mary	Grove	yyy
234	Mazda	RX7	Blue	abc	Dave	Birch	xxx
234	Mazda	RX7	Blue	bcd	Mary	Grove	yyy

Give relational algebra expressions for each of the following queries specified in English (assuming the three tables, with all rows, of the handout).

1. Write a query that returns the Name and Phone of all Persons owning VRN=123.

natural join

$$\Pi_{\text{Name, Phone}}\left(\sigma_{\text{VRN}=123}\left(\text{Own} \bowtie \text{Person}\right)\right)$$

could have parenthesized individual tables

$$\Pi_{\text{Name, Phone}}\left(\sigma_{\text{VRN}=123}\left(\left(\rho_{\text{Own}(\text{VRN, OSSN})}\ \text{Own}\right) \bowtie_{\text{OSSN}=\text{PSSN}} \left(\rho_{\text{Person}(\text{PSSN, Name, ...})}\ \text{Person}\right)\right)\right)$$

rename SSNs to distinguish them

theta (θ) join makes join condition explicit

This shorthand would be acceptable

$$\Pi_{\text{Name, Phone}}\left(\sigma_{\text{VRN}=123}\left(\text{Own} \bowtie_{\text{O.SSN}=\text{P.SSN}} \text{Person}\right)\right)$$

Push select(s) inward

$$\Pi_{\text{Name, Phone}}\left(\left(\sigma_{\text{VRN}=123}\ \text{Own}\right) \bowtie \text{Person}\right)$$

Generally, "pushing" projections almost never a good idea

$$\Pi_{\text{Name, Phone}}\left(\left(\sigma_{\text{VRN}=123}\ \text{Own}\right) \bowtie \left(\Pi_{\text{Name, Phone}}\ \text{Person}\right)\right)$$ wrong

No basis for join

1.

Own		Person
VRN	SSN	SSN ...
012	abc	def
123	bcd	abc
234	cde	bcd
		cde

$\bowtie$

~~012 abc abc ...~~

123 bcd bcd ...

~~234 cde cde ...~~

$\sigma_{VRN=123}$

---

what if answer had been

$$\pi_{\substack{Name,\\Phone}}\left(\sigma_{VRN=123}\left(Own \times Person\right)\right)$$

VRN	SSN		SSN
012	abc	X	def
123	bcd		abc
234	cde		bcd
			cde

$\Rightarrow$

012 abc def

⋮

123 bcd def
123 bcd abc
123 bcd bcd
123 bcd cde

~all of these would pass select test =123

2. Write a query to show the VRN and Mo of each owned Vehicle and the Name and Addr of the Person who owns it.

V.VRN or O.VRN

$\Pi_{VRN, Mo, Name, Addr}$ ((Vehicle ⋈ Own) ⋈ Person)

$\Pi_{VRN, Mo, Name, Addr}$ ((Vehicle ⋈ (Own ⋈ Person))

$\Pi_{VRN, Mo, Name, Addr}$ ((Vehicle ⋈ Person) ⋈ Own)

and of course, can write natural join as theta join that spells out the join condition

$\bowtie_{V.VRN = O.VRN \wedge P.SSN = O.SSN}$

Shorthand ok

$\Pi_{VRN, Mo, Name, Addr}$ (Vehicle ⋈ Own ⋈ Person)

3. Write a query to ~~to~~ show the VRN and Mo of each owned Vehicle by someone on 'Birch' and the Name of the Person who owns it.

$$\Pi_{VRN, Mo, Name}\left(\sigma_{Addr='Birch'}\left(Vehicle \bowtie Own \bowtie Person\right)\right)$$

$$\Pi_{VRN, Mo, Name}\left(Vehicle \bowtie Own \bowtie \left(\sigma_{Addr='Birch'} Person\right)\right)$$ Push select inward

$$\Pi_{VRN, Mo, Name}\left(Vehicle \bowtie \left(Own \bowtie_{O.SSN=P.SSN \wedge Addr='Birch'} Person\right)\right)$$

This is OK, since ~~$Own \overset{R}{\bowtie}_{\theta} Person \overset{S}{}$~~

$\equiv \sigma_{\theta}(R \times S)$

but I don't like its style, because 'Birch' is a constant and is not an attribute of either table. Addr='Birch' is best thought of as a select condition

4. Write a query that returns the VRN and Mo of any vehicle that isn't owned.

set difference operands must have same format

$$\left(\pi_{\substack{VRN,\\ Mo}} Vehicle\right) - \left(\pi_{\substack{VRN,\\ Mo}} \left(Vehicle \bowtie Own\right)\right)$$

all vehicles — all owned vehicles

all unowned vehicles

$$\pi_{\substack{VRN,\\ Mo}} \left(\left(\pi_{VRN} Vehicle - \pi_{VRN} Own\right) \bowtie Vehicle\right)$$

VRNs of all unowned vehicles

"join back" to get Mo
(this strategy was illustrated in video)

5. Write a query that returns pairs of SSNs for (different) Persons that live at the same Addr.

$$\Pi_{P1.SSN,\ P2.SSN}\left(\left(\rho_{P1(\ldots)}\ Person\right) \bowtie_{\substack{P1.Addr = P2.Addr \\ \wedge\ P1.SSN <> P2.SSN}} \left(\rho_{P2(\ldots)}\ Person\right)\right)$$

// SSN comparison better than name comparison in this case. Why? In fact, Name comparison would be incorrect

OR

$$\begin{cases} P1 := Person \\ P2 := Person \\ \Pi_{P1.SSN,\ P2.SSN}\left(P1 \bowtie_{\substack{P1.Addr = P2.Addr \\ \wedge\ P1.SSN <> P2.SSN}} P2\right) \end{cases}$$

~ what's wrong with natural join here

$$\Pi_{P1.SSN,\ P2.SSN}\left(\sigma_{\substack{P1.SSN \\ <> \\ P2.SSN}}\left(\rho_{P1(\ldots)}\ Person \bowtie_{P1.Addr = P2.Addr} \rho_{P2(\ldots)}\ Person\right)\right)$$

The versions above will return answers of form

P1.SSN	P2.SSN
⋮	⋮
123	456
⋮	⋮
456	123
⋮	⋮

} conceptual duplication

Best to replace '<>' above with '<' (or '>')

only get 123 456

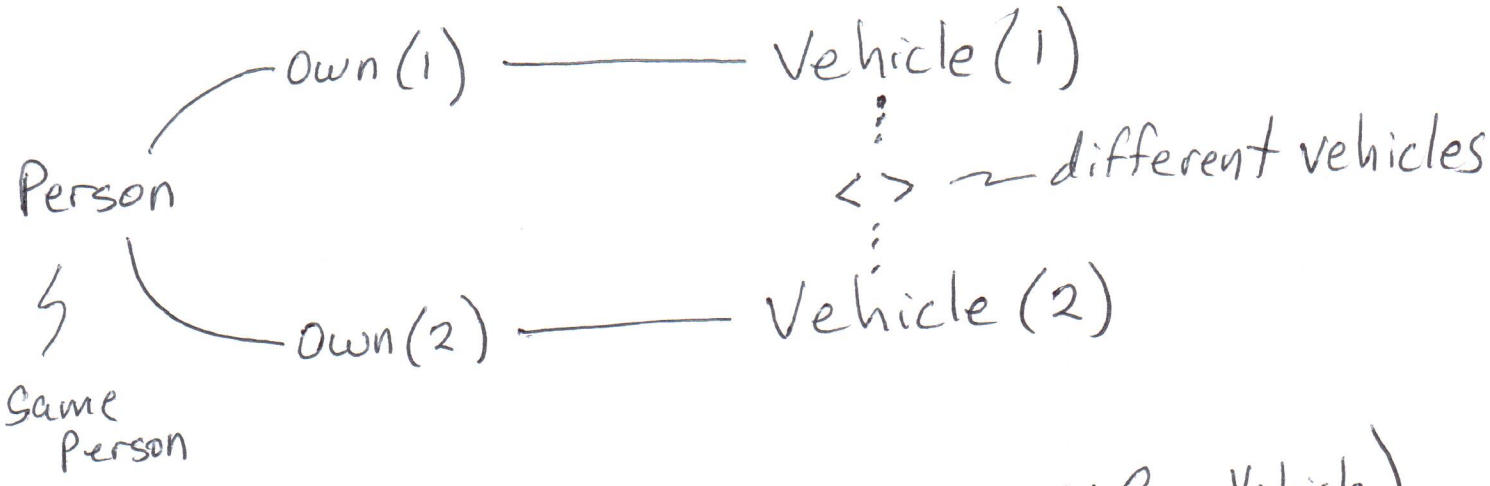
only get 456 123

6. Write a query that returns pairs of Names for (different) Persons that live at the same Addr.

$$\Pi_{P1.Name,\ P2.Name}\left(\left(\rho_{P1(\ldots)}\ Person\right) \bowtie_{P1.Addr = P2.Addr \wedge P1.SSN < P2.SSN} \left(\rho_{P2(\ldots)}\ Person\right)\right)$$

still using SSN in join!
(e.g., my father and I shared the same name)

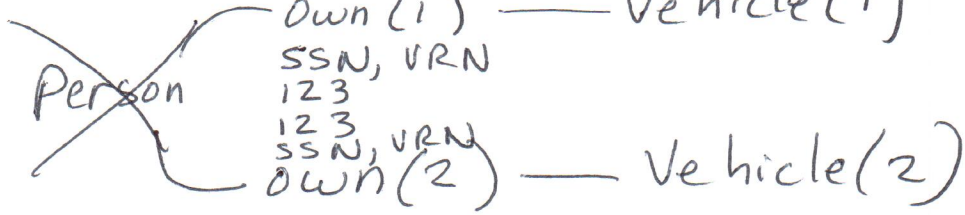
7. Write a query that lists pairs of Vehicles by Ma and Mo that are owned by the same Person.



$$\Pi_{V1.Ma,\ V1.Mo,\ V2.Ma,\ V2.Mo}\left(\rho_{V1(...)}\text{Vehicle} \bowtie_{V1.VRN=O1.VRN} \rho_{O1(...)}\text{Own} \bowtie_{O1.SSN=P.SSN} \text{Person} \bowtie_{P.SSN=O2.SSN \wedge O2.VRN<>O1.VRN} \rho_{O2(...)}\text{Own} \bowtie_{O1.VRN=V2.VRN} \rho_{V2(...)}\text{Vehicle}\right)$$

Can parenthisize & rename in different ways

But bigger observation — don't need to involve Person at all, because Own includes a person's unique ID



$$\Pi_{V1.Ma,\ V1.Mo,\ V2.Ma,\ V2.Mo}\left(\rho_{V1(...)}\text{Vehicle} \bowtie_{V1.VRN=O1.VRN} \rho_{O1(...)}\text{Own} \bowtie_{O1.SSN=O2.SSN \wedge O2.VRN<>O1.VRN} \rho_{O2(...)}\text{Own} \bowtie_{O2.VRN=V2.VRN} \rho_{V2(...)}\text{Vehicle}\right)$$

use < (or >) to get rid of conceptual redundancy

For problems like 7, of non-trivial complexity, its helpful to break them apart

(1) What relations are involved?
Ma, Mo requires that Vehicle be involved.
If only VRNs were needed as the Vehicle info, then only Own would be needed

(2) Decompose problem is often helpful. A simpler problem would be to list pairs of vehicles owned by same person

$$\rho_{O1(\ldots)}\,Own \bowtie_{\substack{O1.SSN = O2.SSN \\ \wedge\ O1.VRN < O2.VRN}} \rho_{O2(\ldots)}\,Own$$

then recognize that Ma, Mo required too, which implicates Vehicle

or perhaps start with simpler problem of listing pairs of all different vehicles

$$\rho_{V1(\ldots)}\,Vehicle \bowtie_{V1.VRN < V2.VRN} \rho_{V2(\ldots)}\,Vehicle$$

and build on that

8. Write a query that lists pairs of Persons by SSN and Name that own a Vehicle of the same Ma, Mo and Color.

Its possible for same person to have two different vehicles with same Ma, Mo, Color; how would query change if allowed

but don't have to be same vehicle (& would not be in my interpretation, but in some settings the DB may allow for two owners of same vehicle (co-owners)

$$\Pi_{\substack{P1.SSN,\\ P1.Name,\\ P2.SSN,\\ P2.Name}} \Big(\rho_{P1(\ldots)}\,\text{Person} \underset{\substack{P1.SSN\\ <P2.SSN}}{\bowtie} \rho_{P2(\ldots)}\,\text{Person} \quad \text{— different persons}$$

$$\underset{\substack{P1.SSN\\ =O1.SSN}}{\bowtie} \rho_{O1(\ldots)}\,\text{Own} \underset{\substack{P2.SSN\\ =O2.SSN}}{\bowtie} \rho_{O2(\ldots)}\,\text{Own}$$

$$\underset{\substack{O1.VRN\\ =V1.VRN}}{\bowtie} \rho_{V1(\ldots)}\,\text{Vehicle}$$

$$\underset{\substack{O2.VRN\\ =V2.VRN}}{\bowtie} \rho_{V2(\ldots)}\,\text{Vehicle} \Big)$$

$$\wedge\ V1.Ma = V2.Ma$$

$$\wedge\ V1.Mo = V2.Mo$$

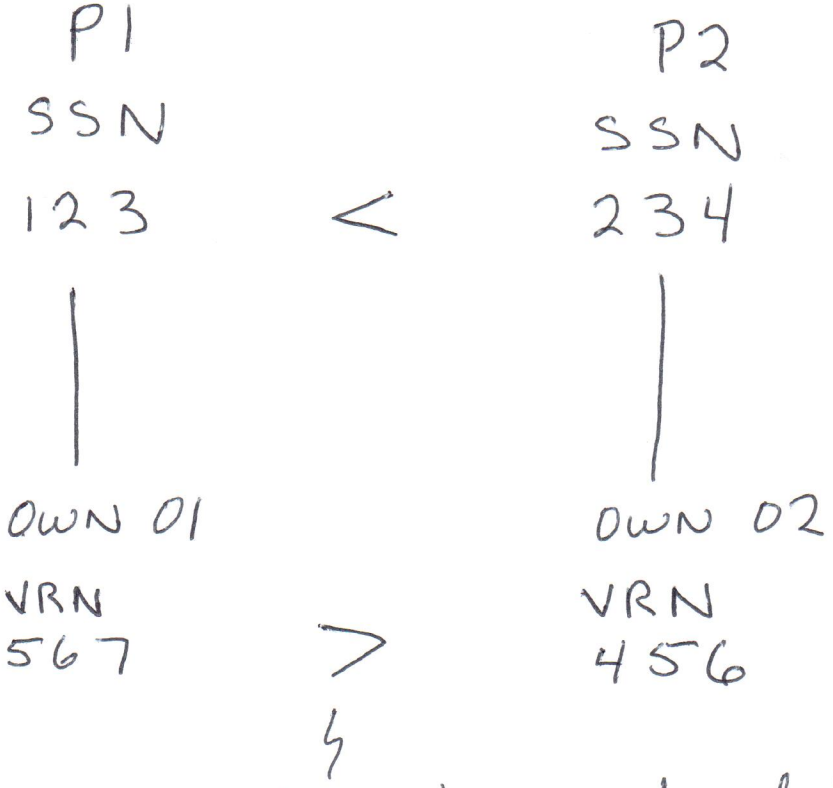
$$\wedge\ V1.Color = V2.Color$$

$$(\wedge\ V1.VRN < V2.VRN) \quad \text{or } <> ?$$

— if we want to distinguish different vehicles (I think we do)

Notice that renaming in inner scope is ok to reference in outer scope. If we wanted to disallow then put all renames in outermost scope

$$\rho_{P1\ldots}\rho_{P2\ldots}\rho_{O1\ldots}\rho_{O2\ldots}\rho_{V1\ldots}\rho_{V2\ldots}\left(\Pi_{P1.SSN,\ldots}(\ldots)\right)$$



if we have ordered by SSN, then ordering by VRN may cause conflict

use <> in comparing VRNs for &

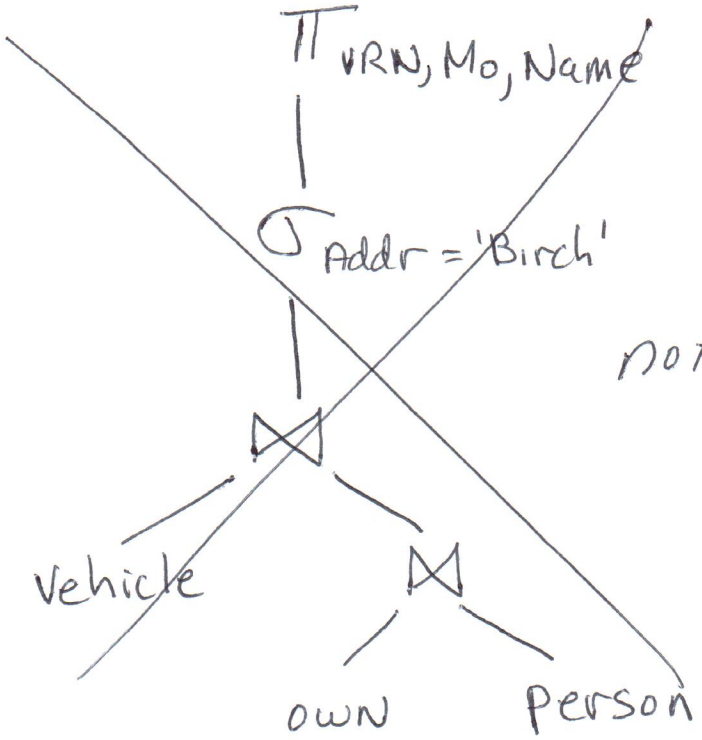
9. Suppose that there was a Price attribute on Vehicle. Write a query in which each car is listed (paired) with each other car that costs less than it does.

$$\Pi_{V1.VRN,\ V2.VRN}\left(\rho_{V1(\ldots)}\text{Vehicle} \bowtie_{V1.Price < V2.Price} \rho_{V2(\ldots)}\text{Vehicle}\right)$$

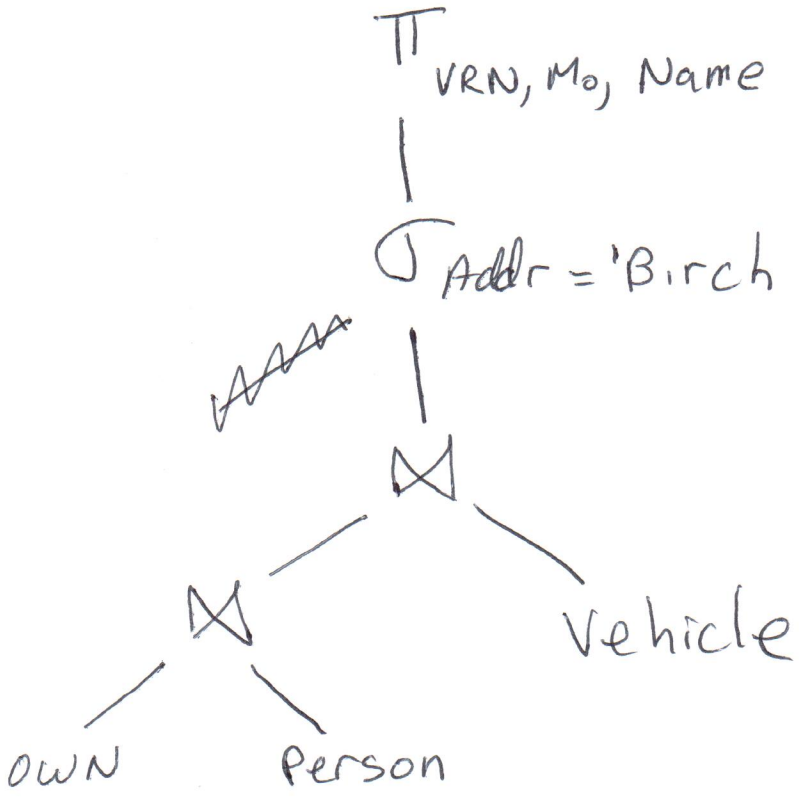
5

not all joins are equality joins

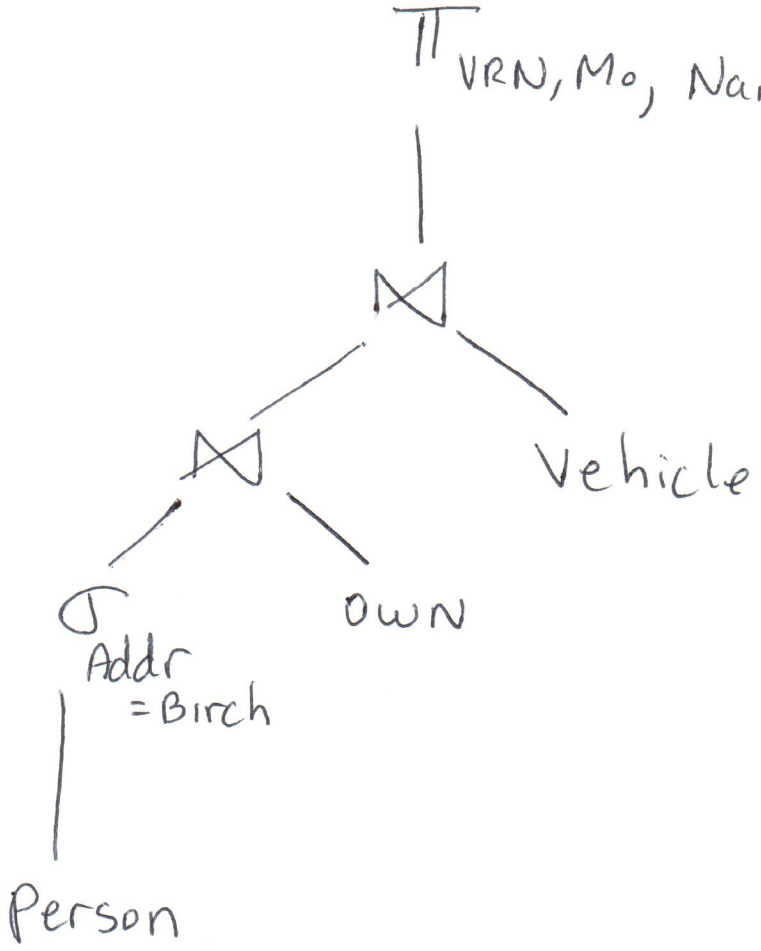
10. Give a left-deep expression tree for the relational algebra query specification of question 3. Your expression tree need not correspond precisely to your answer for question 3.



not left deep



left deep



left deep

others ~~a~~possible