CS 3265 and CS 5265
Vanderbilt University

Lecture on Relational Algebra

This lecture assumes that you have

- Watched videos from WSPDBC DB1 Introduction and Relational Databases
- Watched videos from WSPDBC DB4 Relational Algebra

or

- Read sections 1.1-1.2 and 2.2-2.3 of U/W
- Read sections 2.3 and 2.4 of U/W

WSPDBC: Jennifer Widom's Self-Paced Database mini-courses, offered by Stanford University as an online reference (http://online.stanford.edu/course/databases-self-paced)

U/W: A First Course in Database Systems, 3rd edition, by Ullman and Widom, U/W

Douglas H. Fisher

Rosling visualization designs

Candidate Design 1

TimeStampedCountry (CountryName, Year, Population, AveLifeExpect, AveIncome)

Candidate Design 2

TimeStampedCountry (CountryName, Year, Population, AveLifeExpect, AveIncome, Continent)

Candidate Design 3

TimeStampedCountry (CountryName, Year, Population, AveLifeExpect, AveIncome)
Country (CountryName, Continent)

Candidate Design 4

TimeStampedRegion (RegionName, Year, Population, AveLifeExpect, AveIncome)
Region (SubordinateRegionName, SuperordinateRegionName)

Candidate Design 5

TimeStampedRegion (RegionName, Year, Population, AveLifeExpect, AveIncome)
Region (SubordinateRegionName, SuperordinateRegionName)
Country (CountryName)
Continent(CountinentName)
State(StateName)

Still other designs, perhaps computing averages from finer grained data

Consider candidate 5 of the alternative Rosling visualization designs.

TimeStampedRegion (RegionName, Year, Population, AveLifeExpect, AveIncome)
Region (SubordinateRegionName, SuperordinateRegionName)
Country (CountryName)
Continent(ContinentName)
State(StateName)

Give a relational algebra query that lists each region (by RegionName) with its (immediate) "parent" superordinate region (by RegionName), and in separate rows of the result, lists the regions (by name) with their "grandparent" superordinate regions (by name).

Consider candidate 5 of the alternative Rosling visualization designs.

TimeStampedRegion (RegionName, Year, Population, AveLifeExpect, AveIncome)
Region (SubordinateRegionName, SuperordinateRegionName)
Country (CountryName)
Continant(ContinentName)
State(StateName)

Give a relational algebra query that lists each region (by RegionName) with its (immediate) "parent" superordinate region (by RegionName), and in separate rows of the result, lists the regions (by name) with their "grandparent" superordinate regions (by name).

This is simply the Region relation itself, since Region pairs each region and its parent. Write this as

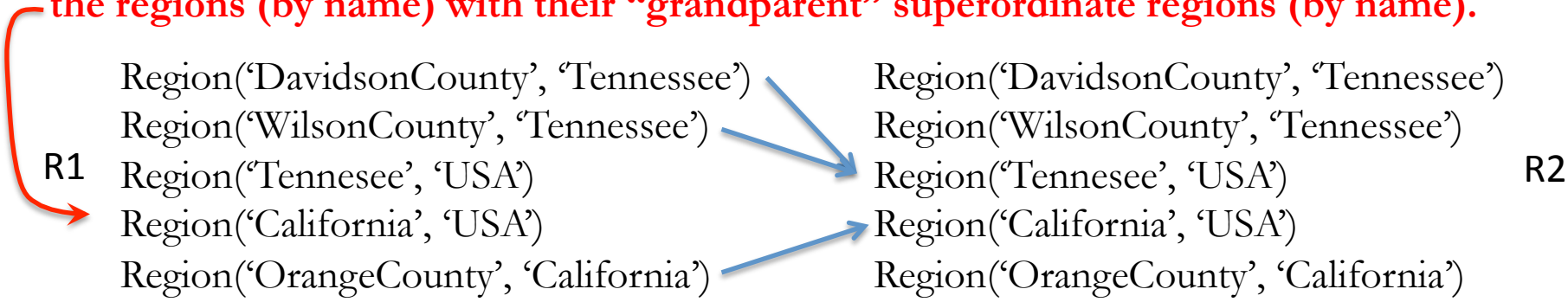
Its ok to abbreviate if its crystal clear what is intended

$\pi_{\text{SubName, SupName}}$ (Region) or simply write as Region

Consider candidate 5 of the alternative Rosling visualization designs.

TimeStampedRegion (RegionName, Year, Population, AveLifeExpect, AveIncome)
Region (SubordinateRegionName, SuperordinateRegionName)
Country (CountryName)
Continent(ContinentName)
State(StateName)

Give a relational algebra query that lists each region (by RegionName) with its (immediate) "parent" superordinate region (by RegionName), and in separate rows of the result, lists **the regions (by name) with their "grandparent" superordinate regions (by name).**



Join Region with itself under condition of R1.SupName = R2.SubName (why R1, R2?)

theta

$\rho_{R1(\ldots)}(\text{Region}) \text{ join}_{R1.SupName = R2.SubName} \rho_{R2(\ldots)}(\text{Region})$

Or you could write

R1 := Region; R2 := Region; R1 join $_{R1.SupName = R2.SubName}$ R2

Consider candidate 5 of the alternative Rosling visualization designs.

TimeStampedRegion (RegionName, Year, Population, AveLifeExpect, AveIncome)
Region (SubordinateRegionName, SuperordinateRegionName)
Country (CountryName)
Continent(ContinentName)
State(StateName)

Give a relational algebra query that lists each region (by RegionName) with its (immediate) "parent" superordinate region (by RegionName), and in separate rows of the result, lists the regions (by name) with their "grandparent" superordinate regions (by name).

Putting the two parts together

$$\text{Region} \cup ((\rho_{R1(\ldots)}(\text{Region})) \text{ join}_{R1.\text{SupName} = R2.\text{SubName}} (\rho_{R2(\ldots)}(\text{Region}))) \quad ???$$

Consider candidate 5 of the alternative Rosling visualization designs.

TimeStampedRegion (RegionName, Year, Population, AveLifeExpect, AveIncome)
Region (SubordinateRegionName, SuperordinateRegionName)
Country (CountryName)
Continent(ContinentName)
State(StateName)

Give a relational algebra query that lists each region (by RegionName) with its (immediate) "parent" superordinate region (by RegionName), and in separate rows of the result, lists the regions (by name) with their "grandparent" superordinate regions (by name).

Putting the two parts together

Region U ((ρ $_{R1(\ldots)}$(Region)) join $_{R1.SupName = R2.SubName}$ (ρ $_{R2(\ldots)}$(Region))) No

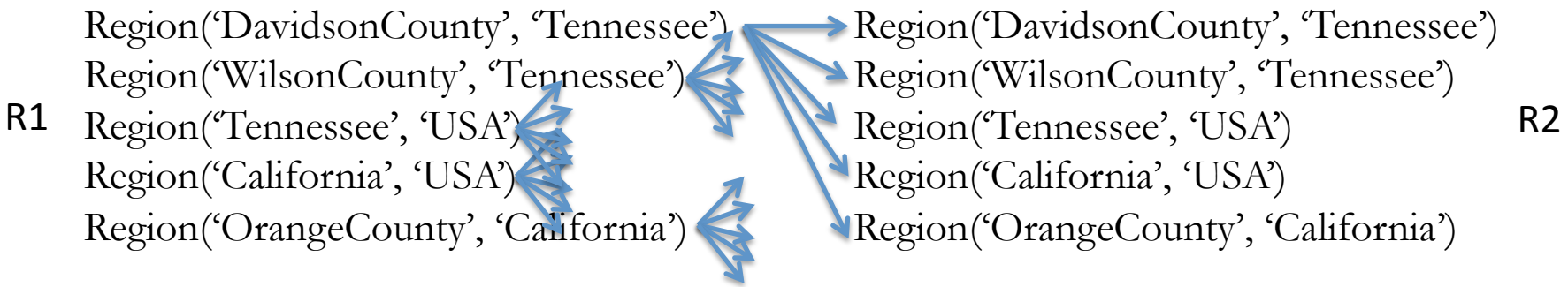
('DavidsonCounty', 'Tennessee') ('DavidsonCounty', 'Tennessee', 'Tennessee', 'USA')

Region U (π $_{R1.SubName,\ R2.SupName}$ ((ρ $_{R1(\ldots)}$(Region)) join $_{R1.SupName = R2.SubName}$ (ρ $_{R2(\ldots)}$(Region))))

Returning to "part 2" of the original problem specification

Give a relational algebra query that lists each region (by RegionName) with its (immediate) "parent" superordinate region (by RegionName), and in separate rows of the result, lists the regions (by name) with their "grandparent" superordinate regions (by name).

Remember that R1 join_Θ R2 = σ_Θ (R1 X R2)

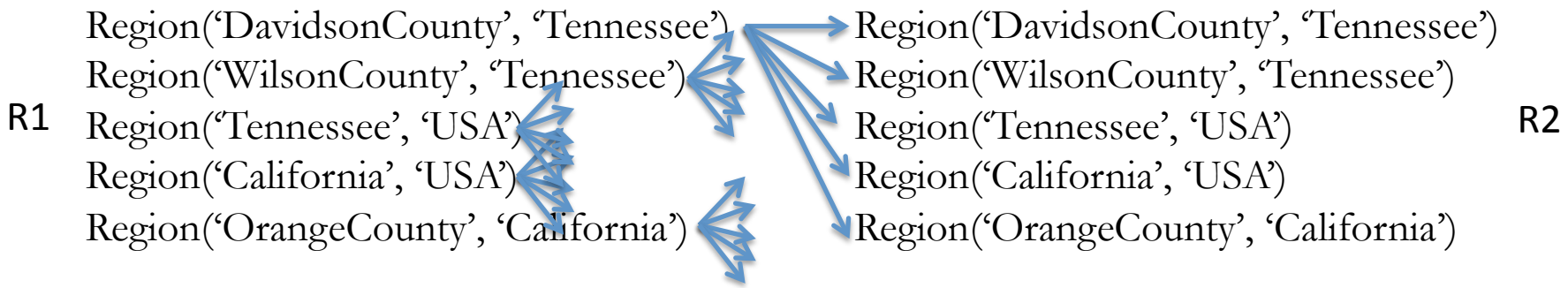


X

('DavidsonCounty', 'Tennessee')	('DavidsonCounty', 'Tennessee')
('DavidsonCounty', 'Tennessee')	('WilsonCounty', 'Tennessee')
('DavidsonCounty', 'Tennessee')	('Tennessee', 'USA')
('DavidsonCounty', 'Tennessee')	('California', 'USA')
('DavidsonCounty', 'Tennessee')	('OrangeCounty', 'California')
('WilsonCounty', 'Tennessee')	('DavidsonCounty', 'Tennessee')
('WilsonCounty', 'Tennessee')	('WilsonCounty', 'Tennessee')
('WilsonCounty', 'Tennessee')	('Tennessee', 'USA')
…	
('OrangeCounty', 'California')	('California', 'USA')
('OrangeCounty', 'California')	('OrangeCounty', 'California')

Consider candidate 5 of the alternative Rosling visualization designs.
Give a relational algebra query that lists each region (by RegionName) with its (immediate) "parent" superordinate region (by RegionName), and in separate rows of the result, lists the regions (by name) with their "grandparent" superordinate regions (by name).

Remember that R1 join$_\Theta$ R2 = σ_Θ (R1 X R2)



σ_Θ

~~('DavidsonCounty', 'Tennessee') ('DavidsonCounty', 'Tennessee')~~
~~('DavidsonCounty', 'Tennessee') ('WilsonCounty', 'Tennessee')~~
('DavidsonCounty', 'Tennessee') ('Tennessee', 'USA')
~~('DavidsonCounty', 'Tennessee') ('California', 'USA')~~
~~('DavidsonCounty', 'Tennessee') ('OrangeCounty', 'California')~~
~~('WilsonCounty', 'Tennessee') ('DavidsonCounty', 'Tennessee')~~
~~('WilsonCounty', 'Tennessee') ('WilsonCounty', 'Tennessee')~~
('WilsonCounty', 'Tennessee') ('Tennessee', 'USA')
…
('OrangeCounty', 'California') ('California', 'USA')
~~('OrangeCounty', 'California') ('OrangeCounty', 'California')~~

Again, consider candidate 5 of the alternative Rosling visualization designs.

Give a relational algebra query that lists each region (by RegionName) with its (immediate) "parent" superordinate region (by RegionName), and in separate rows of the result, lists the regions (by name) with their "grandparent" superordinate regions (by name).

$$\text{Region} \cup ((\pi_{\text{R1.SubName, R2.SupName}}((\rho_{\text{R1}(\ldots)}(\text{Region})) \text{ join}_{\text{R1.SupName = R2.SubName}} (\rho_{\text{R2}(\ldots)}(\text{Region}))))$$

What rows would be present in the result, if the following rows were among the rows in the Region table?

Region('GreenHills', 'Nashville')
Region('Nashville', 'DavidsonCounty')
Region('DavidsonCounty', 'Nashville')
Region('DavidsonCounty', 'Tennessee')
Region('Tennessee', 'USA')
Region('USA', 'NorthAmerica')
Region('NorthAmerica', 'World')

Again, consider candidate 5 of the alternative Rosling visualization designs.

Give a relational algebra query that lists each region (by RegionName) with its (immediate) "parent" superordinate region (by RegionName), and in separate rows of the result, lists the regions (by name) with their "grandparent" superordinate regions (by name).

$$\text{Region} \cup (\pi_{\text{R1.SubName, R2.SupName}} (\rho_{\text{R1(...)}}(\text{Region}) \text{ join}_{\text{R1.SupName = R2.SubName}} \rho_{\text{R2(...)}}(\text{Region}))$$

What rows would be present in the result, if the following rows were among the rows in the Region table?

Region('GreenHills', 'Nashville')
Region('Nashville', 'DavidsonCounty')
Region('DavidsonCounty', 'Nashville')
Region('DavidsonCounty', 'Tennessee')
Region('Tennessee', 'USA')
Region('USA', 'NorthAmerica')
Region('NorthAmerica', 'World')

These would be some of the rows in the result

('GreenHills', 'Nashville')
('Nashville', 'DavidsonCounty')
('DavidsonCounty', 'Nashville')
('DavidsonCounty', 'Tennessee')
('Tennessee', 'USA')
('USA', 'NorthAmerica')
('NorthAmerica', 'World')

Again, consider candidate 5 of the alternative Rosling visualization designs.

Region U ($\pi_{R1.SubName,\ R2.SupName}$ ($\rho_{R1(\ldots)}$(Region) join $_{R1.SupName = R2.SubName}$ $\rho_{R2(\ldots)}$(Region))

What rows would be present in the result, if the following rows were among the rows in the Region table?

Region('GreenHills', 'Nashville')
Region('Nashville', 'DavidsonCounty')
Region('DavidsonCounty', 'Nashville')
Region('DavidsonCounty', 'Tennessee')
Region('Tennessee', 'USA')
Region('USA', 'NorthAmerica')
Region('NorthAmerica', 'World')

Remaining rows in the result

('GreenHills' 'DavidsonCounty')
('Nashville', 'Nashville')
('Nashville', 'Tennessee')
('DavidsonCounty', 'DavisdsonCounty')
('DavidsonCounty', 'USA')
('Tennessee', 'USA')
('USA', 'World')

How would you eliminate all rows from result where SubName = SupName?

How would you eliminate
all rows from result
where SubName = SupName?

$$\text{Region} \cup (\pi_{\text{R1.SubName, R2.SupName}} (\rho_{\text{R1}(\ldots)}(\text{Region}) \text{ join}_{\substack{\text{R1.SupName = R2.SubName} \\ \text{AND} \\ \text{R1.SubName} \neq \text{R2.SupName}}} \rho_{\text{R2}(\ldots)}(\text{Region}))$$

To be complete

$$\sigma_{\text{SubName} \neq \text{SupName}}(\text{Region}) \cup (\pi_{\text{R1.SubName, R2.SupName}} (\rho_{\text{R1}(\ldots)}(\text{Region}) \text{ join}_{\substack{\text{R1.SupName = R2.SubName} \\ \text{AND} \\ \text{R1.SubName} \neq \text{R2.SupName}}} \rho_{\text{R2}(\ldots)}(\text{Region}))$$

Consider candidate 5 of the alternative Rosling visualization designs.

TimeStampedRegion (RegionName, Year, Population, AveLifeExpect, AveIncome)
Region (SubordinateRegionName, SuperordinateRegionName)
Country (CountryName)
Continent(ContinentName)
State(StateName)

What is the result of NATURAL JOINing TimeStampedRegion and Region?

Consider candidate 5 of the alternative Rosling visualization designs.

TimeStampedRegion (RegionName, Year, Population, AveLifeExpect, AveIncome)
Region (SubordinateRegionName, SuperordinateRegionName)
Country (CountryName)
Continent(ContinentName)
State(StateName)

What is the result of NATURAL JOINing TimeStampedRegion and Region?

It is (TimeStampedRegion X Region) – why?

Consider candidate 5 of the alternative Rosling visualization designs.

TimeStampedRegion (RegionName, Year, Population, AveLifeExpect, AveIncome)
Region (SubordinateRegionName, SuperordinateRegionName)
Country (CountryName)
Continent(ContinentName)
State(StateName)

Give a relational algebra query that lists each country (by CountryName) in Europe, in which the Population of the country has decreased in two consecutive years, together with listing the two years and the Population AveLifeExpect, AveIncome for each of the years. That is, the relational schema of the query result will be

(CountryName,
Year1, Population1, AveLifeExpect1, AveIncome1,
Year2, Population2, AveLifeExpect2, AveIncome2)

Post your answer to Bright Space Discussions

Consider joining TimeStampedRegion to itself (which means you will be using the rename operator), with a conjunctive theta condition that ensures the regions are the same, the years are consecutive, and population decreases from one year to the next. What else must you do?

Query Evaluation Trees using RA

Relational algebra is a formal language

- Used in defining the semantics of SQL
- Used as a conceptual language in query evaluation and optimization algorithms

Consider the following Query in SQL and relational algebra:

SELECT *
FROM Shipped S1, Transactions T1
WHERE S1.TransNumber = T1.TransNumber AND
S1.Isbn = **I1** AND T1.PaymentClearanceDate = **CD**

Associate each transactions with the shipments of that it spawned (with additional requirements on the product purchased and the purchase date).

I1 and **CD** are parameters

$(\sigma_{PCD=CD}((\sigma_{Isbn=I1}(Shipped)) \bowtie Transactions))$

$((\sigma_{Isbn=I1}(Shipped)) \bowtie (\sigma_{PCD=CD}(Transactions)))$

$(\sigma_{Isbn=I1}(Shipped \bowtie (\sigma_{PCD=CD}(Transactions))))$

Other possibilities?

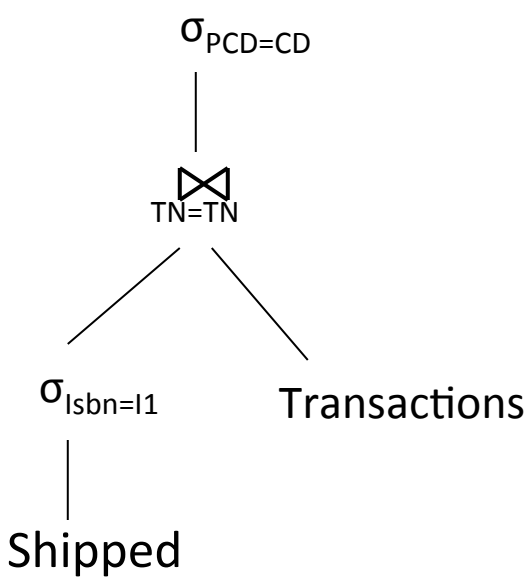
Transaction(TransNumber, PaymentClearanceDate, CustEmailAddr, …)

Shipped(ShipID, Isbn, TransNumber, Quantity, …)

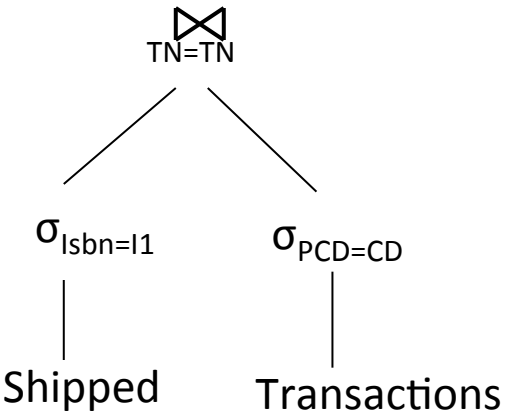
```
SELECT *
FROM Shipped S1, Transactions T1
WHERE S1.TransNumber = T1.TransNumber AND
        S1.Isbn = I1 AND T1.PaymentClearanceDate = CD
```

Query Evaluation Trees

$(\sigma_{PCD=CD}((\sigma_{Isbn=I1}(Shipped)) \bowtie Transactions))$

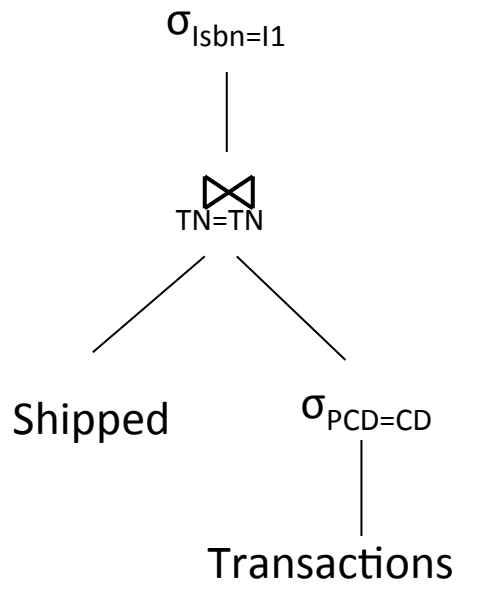


$((\sigma_{Isbn=I1}(Shipped)) \bowtie (\sigma_{PCD=CD}(Transactions)))$



Other trees ??

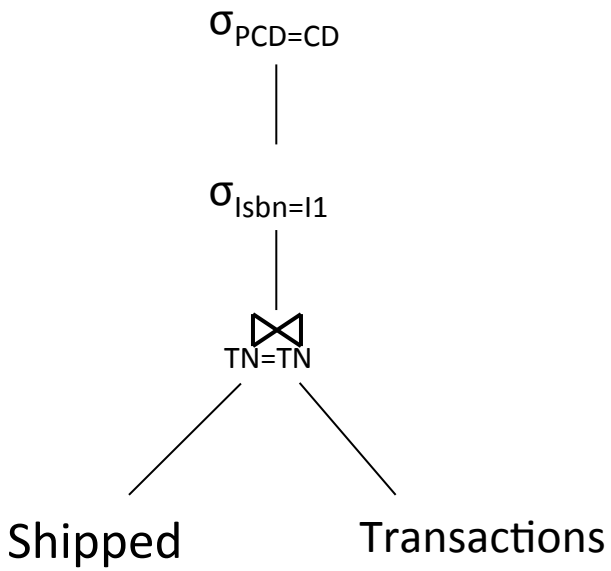
$(\sigma_{Isbn=I1}(Shipped \bowtie (\sigma_{PCD=CD}(Transactions))))$



Left-deep tree: each right child of a join is a base table; **Other left-deep trees?**

```
SELECT *
FROM Shipped S1, Transactions T1
WHERE S1.TransNumber = T1.TransNumber AND
        S1.Isbn = I1 AND T1.PaymentClearanceDate = CD
```

Another left deep query evaluation trees



Select $_{PCD=CD}$ (Select $_{Isbn = I1}$ ((rename $_{S(\ldots)}$ (Shipped)) join $_{S.TN = T.TN}$ (rename $_{T(\ldots)}$ (Transactions))))

Left-deep tree: each right child of a join is a base table

Consider the following Query in SQL and relational algebra:

```
SELECT S1.TransNumber, S2.TransNumber
FROM Shipped S1, Shipped S2, Transactions T1, Transactions T2
WHERE S1.TransNumber = T1.TransNumber AND
        T2.TransNumber = S2.TransNumber AND
        S1.Isbn = I1 AND T1.PaymentClearanceDate = CD AND
        T1.CustomerEmailAddress = T2.CustomerEmailAddress AND
        S2.Isbn = I2
```

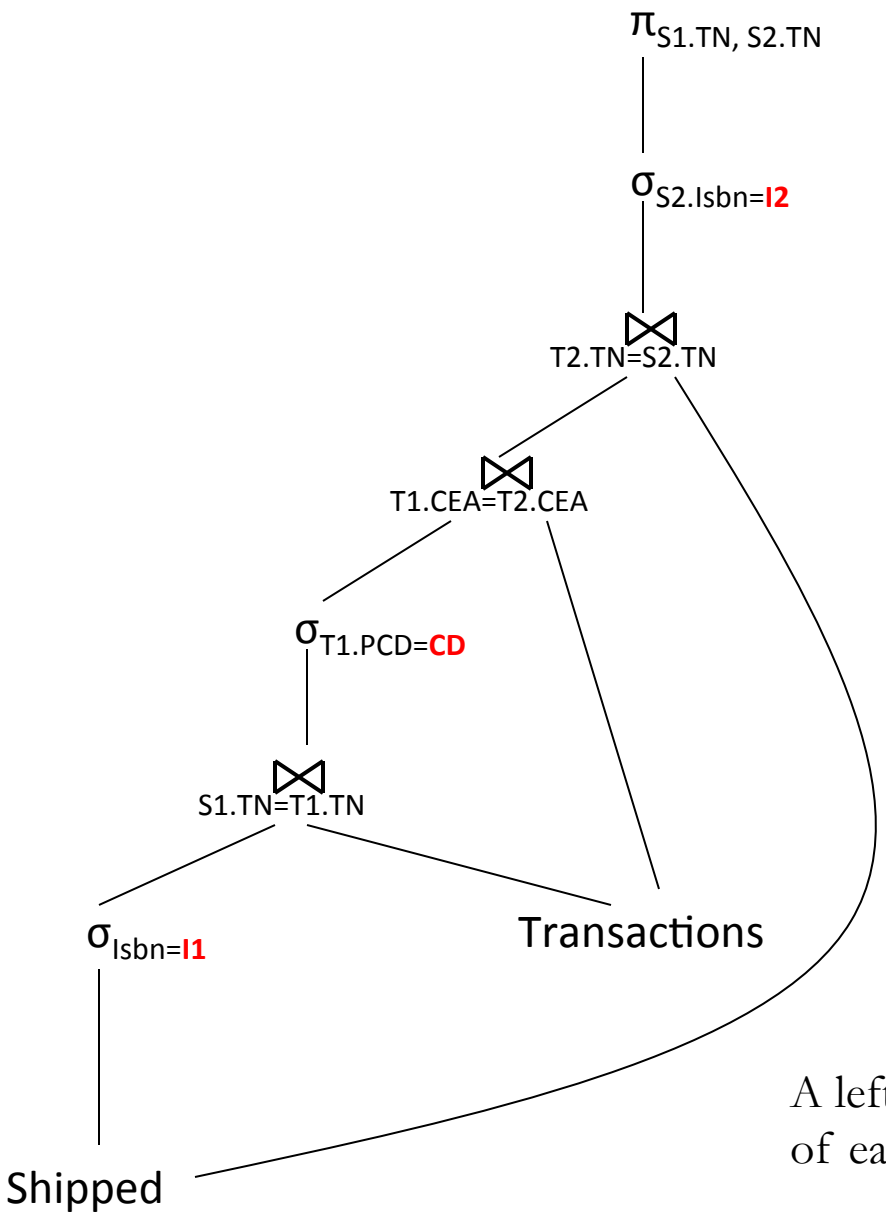
I1, **I2**, and **CD** are parameters

$\pi_{S1.TN,S2.TN}\ (\sigma_{S2.Isbn=I2}\ (\ ((((\sigma_{PCD=CD}\ ((\sigma_{Isbn=I1}\ (\rho_{S1(\ldots)}(Shipped)))) \bowtie (\rho_{T1()},(Transactions)))) \bowtie (\rho_{T2()}(Transactions)))\ ) \bowtie (\rho_{S2()}(Shipped))\ ))$

Please overlook unbalanced parentheses

Project $_{S1.TN,\ S2.TN}$ (Select $_{S2.Isbn=I2}$ (((((Select $_{PCD=CD}$ ((Select $_{Isbn\ =\ I1}$ (rename $_{S1(\ldots)}$ (Shipped)))) join (rename …)

Draw left-deep tree(s) for this query



TN = TransNumber
CEA = CustEmailAddr
PCD = PaymentClearDate
I1, I2, CD are parameters

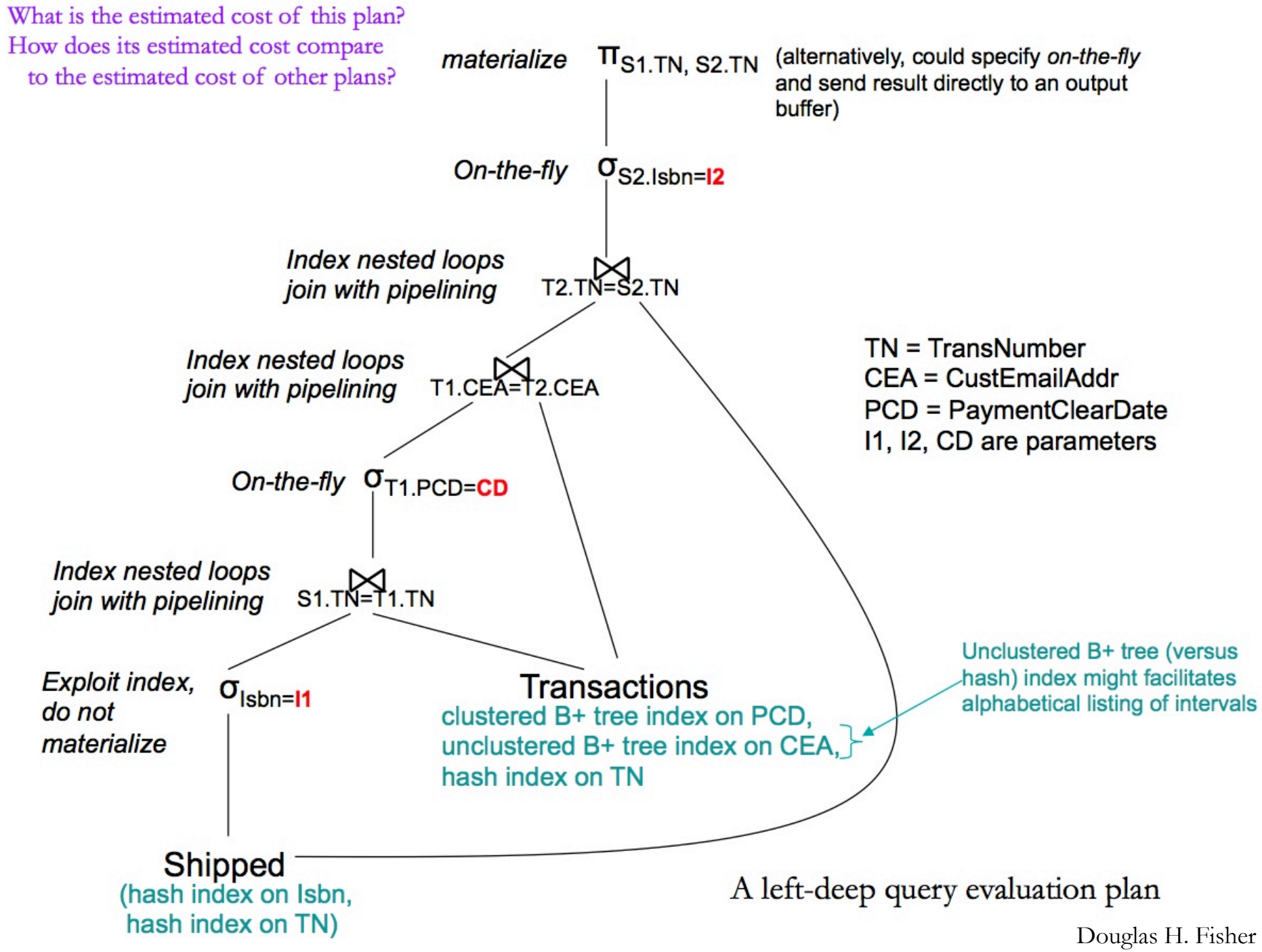
A left-deep query tree: the right child of each join is a base table.

Assume the following conditions hold for a relational DB that we've designed for an e-bookseller.

i) a block/page is 2^12 bytes.
ii) each tuple of Transactions requires 2^4 bytes
iii) each tuple of Shipped requires 2^4 bytes
iv) Each index (for any attribute of any table) requires 2^3 bytes
v) There are 2^27 tuples in Transactions
vi) There are 2^28 tuples in Shipped
vii) There are 2^17 tuples that satisfy PCD=CD
(PCD is PaymentClearanceDate, CD is a particular value, i.e., a constant)
viii) There are 2^20 unique Isbn distributed across Shipped
ix) There are 2^18 unique CEA distributed across Transactions (CEA is CustEmailAddress)
x) clustered B+ tree of order 2^8 index on PCD for Transactions, hash index on TN for Transactions, hash index on CEA for Transactions, hash index on Isbn for Shipped, hash index on TN for Shipped
(TN is TransactionNumber)

• Which of these, (i) – (x), would be stored in the System Catalog. Elaborate as necessary with page references. I am particularly curious about (vii).

• Under the conditions listed above, what is the shallowest that the B+ tree on PCD can possibly be? What is deepest that it can be? Give your answers in terms of index nodes (root included) only (i.e., do not count the data pages as part of the tree).



A left-deep query evaluation plan