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).

Consider the following Query in SQL and relational algebra:

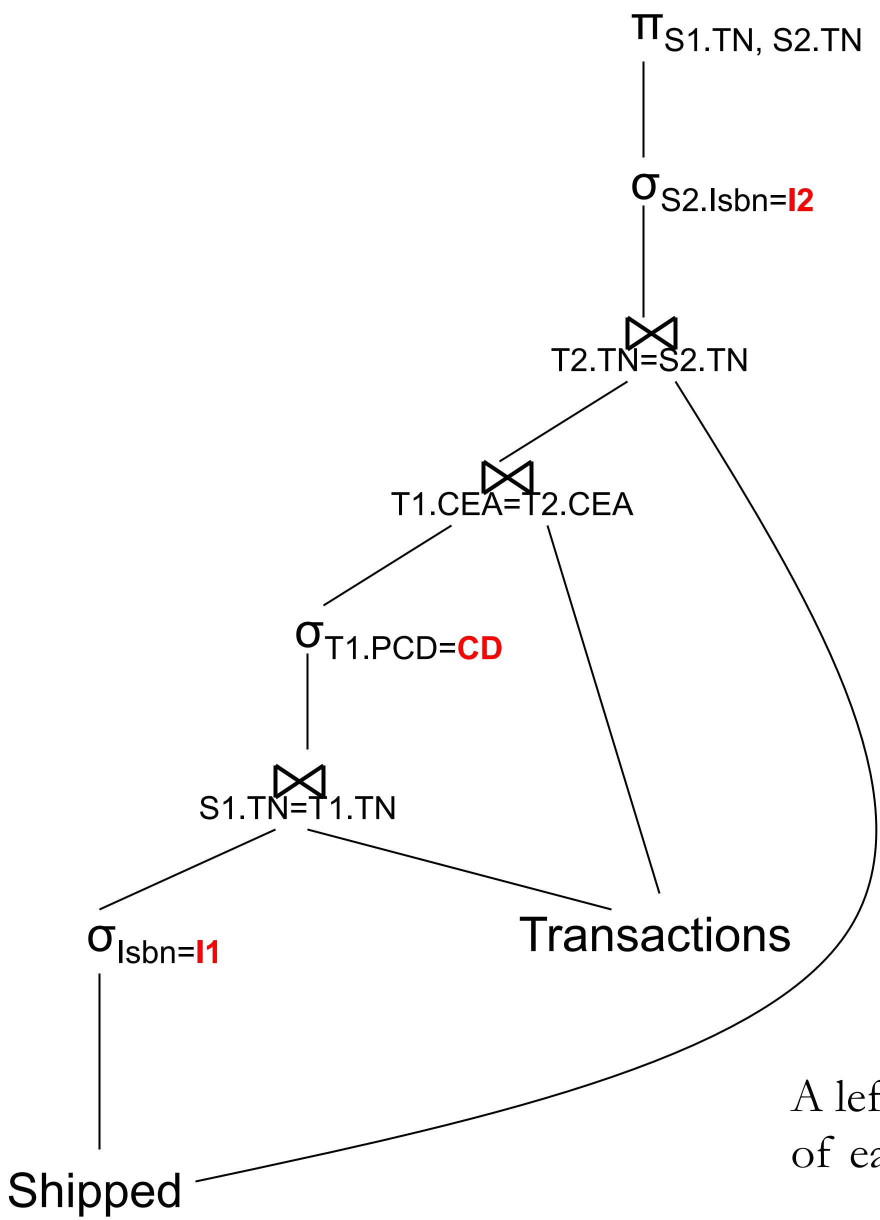
For each book, I1, bought on date CD, by a customer T1.CEA on transaction S1.TN, list the Transactions S2.TN for which T1.CEA bought a second book, I2. (this query might be an auxiliary/nested query for updating CoBought books or the like)

```
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

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

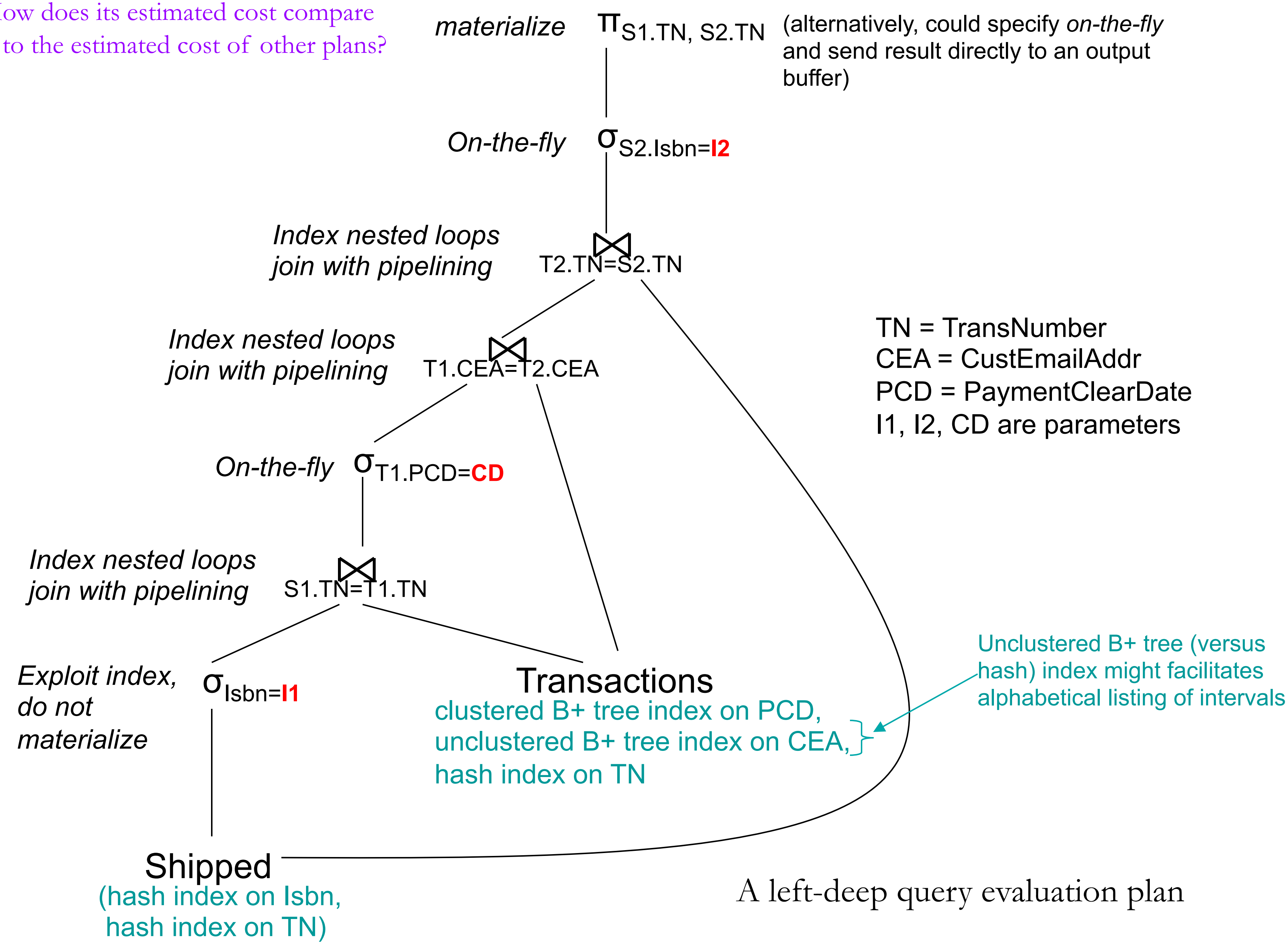
Draw left-deep tree(s) for this query



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.

What is the estimated cost of this plan?
How does its estimated cost compare to the estimated cost of other plans?



A left-deep query evaluation plan

Assume:

- a block/page is 2^{12} bytes (upper range)

Information found in System Catalog

- each tuple of Shipped relation/table requires 2^4 bytes
  - ➔ one block/page holds $2^{12}/2^4 = 2^8$ Shipped tuples
- each index on Isbn of form <Isbn, <pageid, slot#>> requires 2^3 bytes
  - ➔ each block/page holds $2^{12}/2^3 = 2^9$ indices

- there are 2^{28} tuples in Shipped (*Cardinality*) ➔ $2^{28}/2^8 = 2^{20}$ pages <= *Size* <= $2^{21} = 2^{28}/2^7$ pages
- there are 2^{20} distinct Isbns in Shipped (*Index Cardinality*) ➔ $2^{28}/2^9 = 2^{19}$ <= Index Size <= $2^{20} = 2^{28}/2^8$

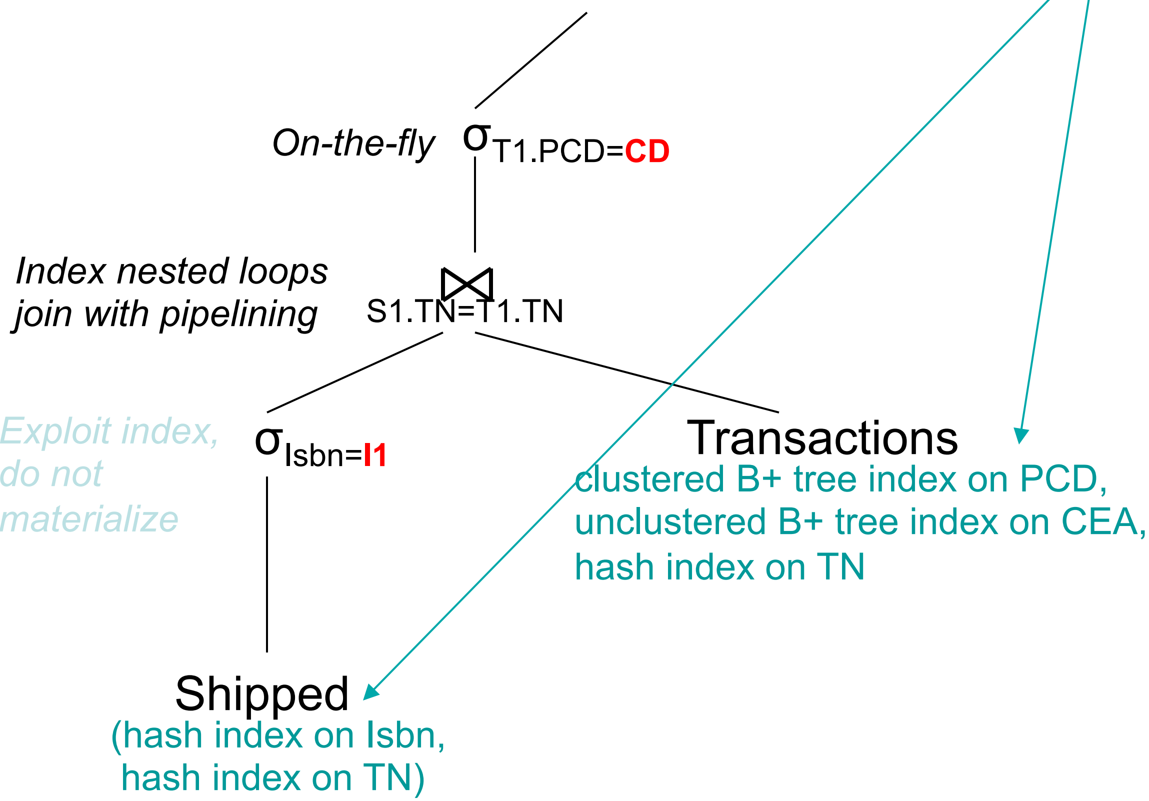
1. Estimate size of result (under *uniform assumption*).

$2^{28}/2^{20} = 2^8$ tuples estimated to satisfy S.Isbn=I1

Estimated size of result = 2^8 tuples

$2^8/2^{28}$ < 5% of Shipped table (probably cheaper to use index, versus file scan, p. 401)

Exploit index, do not materialize

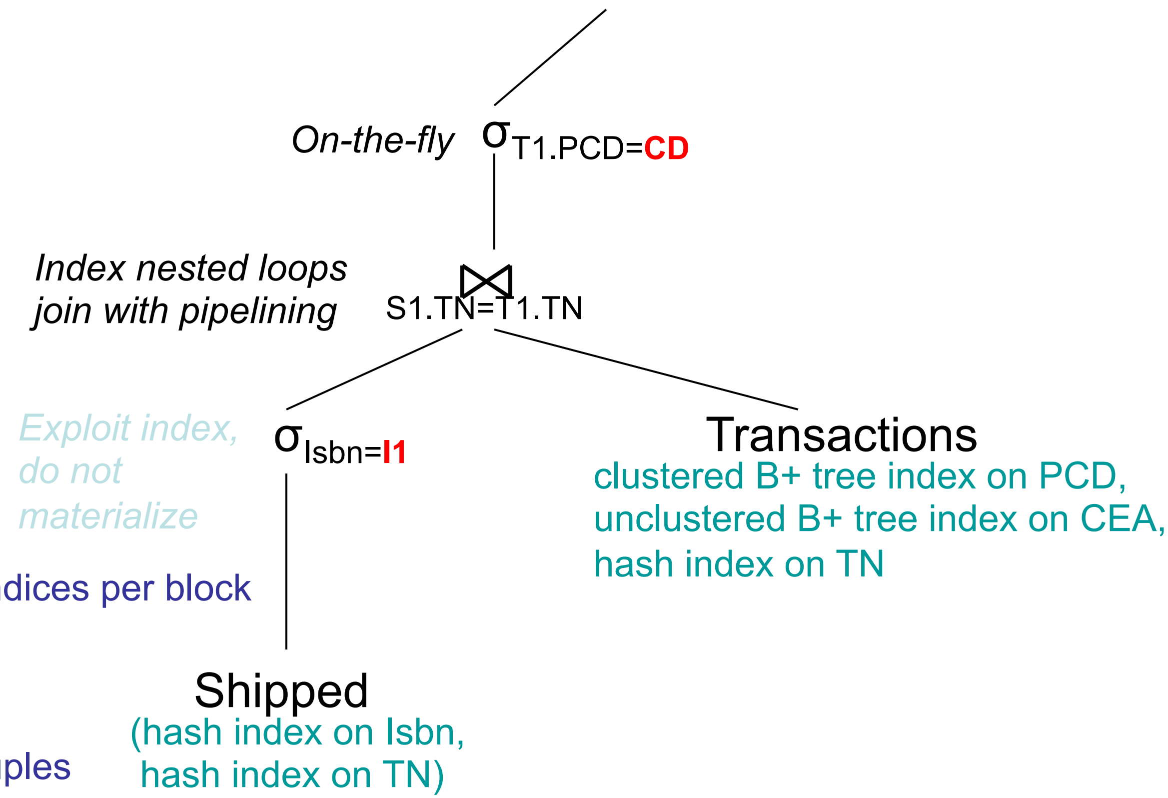


Assume:

- a block/page is 2^{12} bytes (upper range)

- each tuple of Shipped relation/table requires 2^4 bytes
  - ➔ one block/page holds $2^{12}/2^4 = 2^8$ Shipped tuples
- each index on Isbn of form <Isbn, <pageid, slot#>> requires 2^3 bytes
  - ➔ each block/page holds $2^{12}/2^3 = 2^9$ indices

- there are 2^{28} tuples in Shipped (*Cardinality*) ➔ $2^{28}/2^8$ = $\underline{2^{20} \text{ pages} <= \textit{Size} <= 2^{21}}$ = $2^{28}/2^7$ pages
- there are 2^{20} distinct Isbns in Shipped (*Index Cardinality*) ➔ $2^{28}/2^9$ = $\underline{2^{19} <= \text{Index Size} <= 2^{20}}$ = $2^{28}/2^8$



1. Estimate size of result (under *uniform assumption, p. 401*).

$2^{28}/2^{20} = 2^8$ tuples estimated to satisfy S.Isbn=I1

Estimated size of result = 2^8 tuples

2. Estimate # of page scans using Index on Isbn

1 index page since 2^8 per Isbn < 2^9 indices per block

between 1 data page (if all 2^8 tuples fit on 1 page) *and* 2^8 data pages (if each 2^8 tuples on different data page)

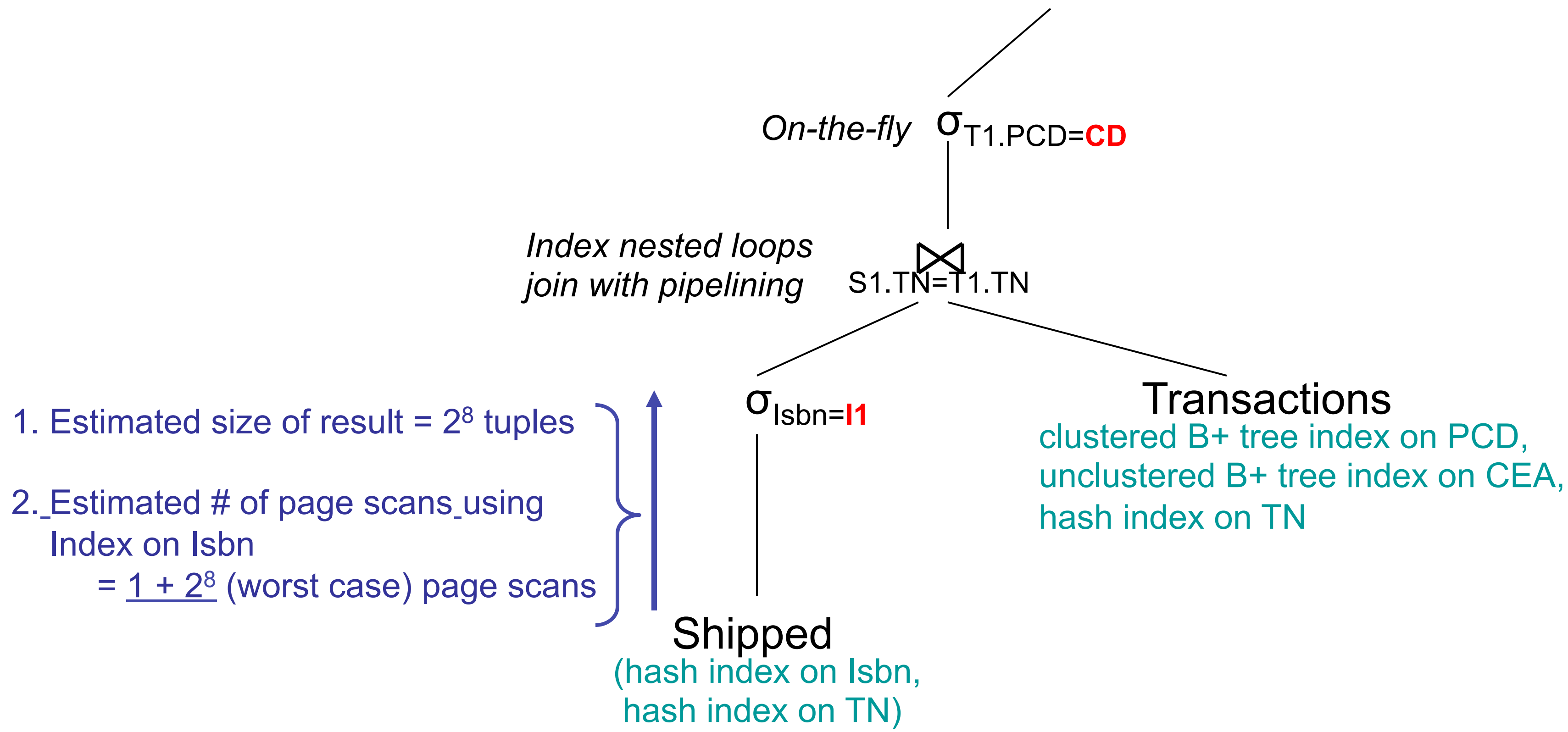
Exercise: can you find some reference to an “average” or expected number of data pages?

Assume:

- a block/page is 2^{12} bytes (upper range)

- each tuple of Shipped relation/table requires 2^4 bytes
  - ➔ one block/page holds $2^{12}/2^4 = 2^8$ Shipped tuples
- each index on Isbn of form <Isbn, <pageid, slot#>> requires 2^3 bytes
  - ➔ each block/page holds $2^{12}/2^3 = 2^9$ indices

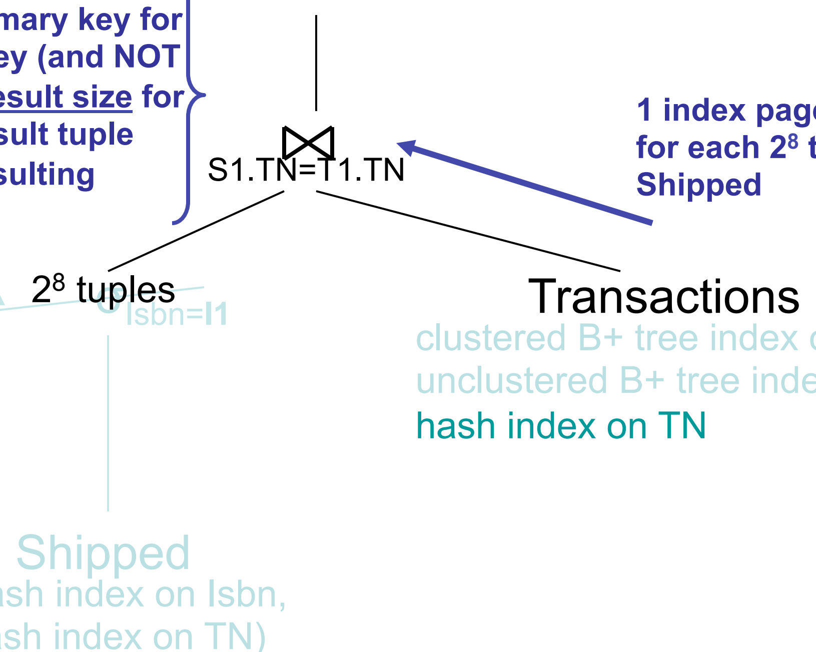
- there are 2^{28} tuples in Shipped (*Cardinality*) ➔ $2^{28}/2^8 = \underline{2^{20} \text{ pages} <= \textit{Size} <= 2^{21}} = 2^{28}/2^7$ pages
- there are 2^{20} distinct Isbns in Shipped (*Index Cardinality*) ➔ $2^{28}/2^9 = \underline{2^{19} <= \text{Index Size} <= 2^{20}} = 2^{28}/2^8$



Assume:

- a block/page is 2^{12} bytes (upper range)
- each tuple of Shipped relation/table requires 2^4 bytes
  ➔ one block/page holds $2^{12}/2^4 = 2^8$ Shipped tuples
- each index on Isbn of form <Isbn, <pageid, slot#>> requires 2^3 bytes
  ➔ each block/page holds $2^{12}/2^3 = 2^9$ indices
- there are 2^{28} tuples in Shipped (*Cardinality*) ➔ $2^{28}/2^8$ = $\underline{2^{20} \text{ pages} <= \textit{Size} <= 2^{21}}$ = $2^{28}/2^7$ pages
- there are 2^{20} distinct Isbns in Shipped (*Index Cardinality*) ➔ $2^{28}/2^9$ = $\underline{2^{19} <= \text{Index Size} <= 2^{20}}$ = $2^{28}/2^8$

In general, a join can increase or decrease the number of tuples, but TN is the primary key for Transactions and TN is a foreign key (and NOT NULL) for Shipped, so <u>estimated result size</u> for join remains <u>2^8 tuples</u> (but each result tuple is about twice the size of tuples resulting from initial select)



1 index page and 1 data page for each 2^8 tuples from σ on Shipped

1. Estimated size of result = 2^8 tuples
2. Estimated # of page scans using Index on Isbn
   = $\underline{1 + 2^8}$ (worst case) page scans

Estimate the expected result size and worst case page scans for this operation.
What additional informaion do you need to know?

On-the-fly $\sigma_{T1.PCD=CD}$

(exercise: can you find a reference to lower expected cost stemming from possibility of Transaction Index or data pages being in page buffer?)

In general, a join can increase or decrease the number of tuples, but TN is the primary key for Transactions and TN is a foreign key (and NOT NULL) for Shipped, so expected result size for join remains <u>2^8 tuples</u> (but each result tuple is about twice the size of tuples resulting from initial select)

2^8 tuples

$\bowtie_{S1.TN=T1.TN}$

$(1+1)2^8 = \underline{2^9}$ page scans (worst case)

$\sigma_{Isbn=I1}$

Transactions
clustered B+ tree index on PCD,
unclustered B+ tree index on CEA,
hash index on TN

Estimated # of page scans using
Index on Isbn
= $\underline{1 + 2^8}$ (worst case) page scans

Shipped
(hash index on Isbn,
hash index on TN)

Total estimated page scans so far:
$1 + 2^8 + 2^9$

1. Finish estimating the total cost of the example plan (found on slide 3).

2. Give 2 alternative left deep plans for the sample query.

3. Estimate the cost of these alternative left deep plans (remember: the index and other catalog assumptions will remain the same!!)