CSE 344

APRIL 25TH – DISK I/O

ADMINISTRIVIA

- **HW4 Due Tonight**
- **OQ5 Due Tonight**
- **HW5 Out Tonight**
  - SQL++
  - Due next Wednesday, 11:30

Student

ID	fName	lName
10	Tom	Hanks
20	Amy	Hanks
…		

WHICH INDEXES?

The *index selection problem*

- Given a table, and a “workload” (big Java application with lots of SQL queries), decide which indexes to create (and which ones NOT to create!)

Who does index selection:

- The database administrator DBA
- Semi-automatically, using a database administration tool

INDEX SELECTION: WHICH SEARCH KEY

Make some attribute K a search key if the WHERE clause contains:

- An exact match on K
- A range predicate on K
- A join on K

THE INDEX SELECTION PROBLEM 1

```
V(M, N, P);
```

Your workload is this

100000 queries:

```
SELECT *
FROM V
WHERE N=?
```

100 queries:

```
SELECT *
FROM V
WHERE P=?
```

THE INDEX SELECTION PROBLEM 1

```
V(M, N, P);
```

Your workload is this

100000 queries:

```
SELECT *
FROM V
WHERE N=?
```

100 queries:

```
SELECT *
FROM V
WHERE P=?
```

What indexes ?

THE INDEX SELECTION PROBLEM 1

```
V(M, N, P);
```

Your workload is this

100000 queries:

```
SELECT *
FROM V
WHERE N=?
```

100 queries:

```
SELECT *
FROM V
WHERE P=?
```

A: V(N) and V(P) (hash tables or B-trees)

THE INDEX SELECTION PROBLEM 2

```
V(M, N, P);
```

Your workload is this

100000 queries:

```
SELECT *
FROM V
WHERE N>? and N<?
```

100 queries:

```
SELECT *
FROM V
WHERE P=?
```

100000 queries:

```
INSERT INTO V
VALUES (?, ?, ?)
```

What indexes ?

THE INDEX SELECTION PROBLEM 2

```
V(M, N, P);
```

Your workload is this

100000 queries:

```
SELECT *
FROM V
WHERE N>? and N<?
```

100 queries:

```
SELECT *
FROM V
WHERE P=?
```

100000 queries:

```
INSERT INTO V
VALUES (?, ?, ?)
```

A: definitely V(N) (must B-tree); unsure about V(P)

THE INDEX SELECTION PROBLEM 3

```
V(M, N, P);
```

Your workload is this

100000 queries:

```
SELECT *
FROM V
WHERE N=?
```

1000000 queries:

```
SELECT *
FROM V
WHERE N=? and P>?
```

100000 queries:

```
INSERT INTO V
VALUES (?, ?, ?)
```

What indexes ?

THE INDEX SELECTION PROBLEM 3

```
V(M, N, P);
```

Your workload is this

100000 queries:

```
SELECT *
FROM V
WHERE N=?
```

1000000 queries:

```
SELECT *
FROM V
WHERE N=? and P>?
```

100000 queries:

```
INSERT INTO V
VALUES (?, ?, ?)
```

A: V(N, P)

How does this index differ from:

1. Two indexes V(N) and V(P)?
2. An index V(P, N)?

THE INDEX SELECTION PROBLEM 4

V(M, N, P);

Your workload is this

1000 queries:

```
SELECT *
FROM V
WHERE N>? and N<?
```

100000 queries:

```
SELECT *
FROM V
WHERE P>? and P<?
```

What indexes ?

THE INDEX SELECTION PROBLEM 4

```
V(M, N, P);
```

Your workload is this

1000 queries:

```
SELECT *
FROM V
WHERE N>? and N<?
```

100000 queries:

```
SELECT *
FROM V
WHERE P>? and P<?
```

A: V(N) unclustered, V(P) clustered index

TWO TYPICAL KINDS OF QUERIES

```
SELECT *
FROM Movie
WHERE year = ?
```

- Point queries
- What data structure should be used for index?

```
SELECT *
FROM Movie
WHERE year >= ? AND
          year <= ?
```

- Range queries
- What data structure should be used for index?

BASIC INDEX SELECTION GUIDELINES

Consider queries in workload in order of importance

Consider relations accessed by query

- No point indexing other relations

Look at WHERE clause for possible search key

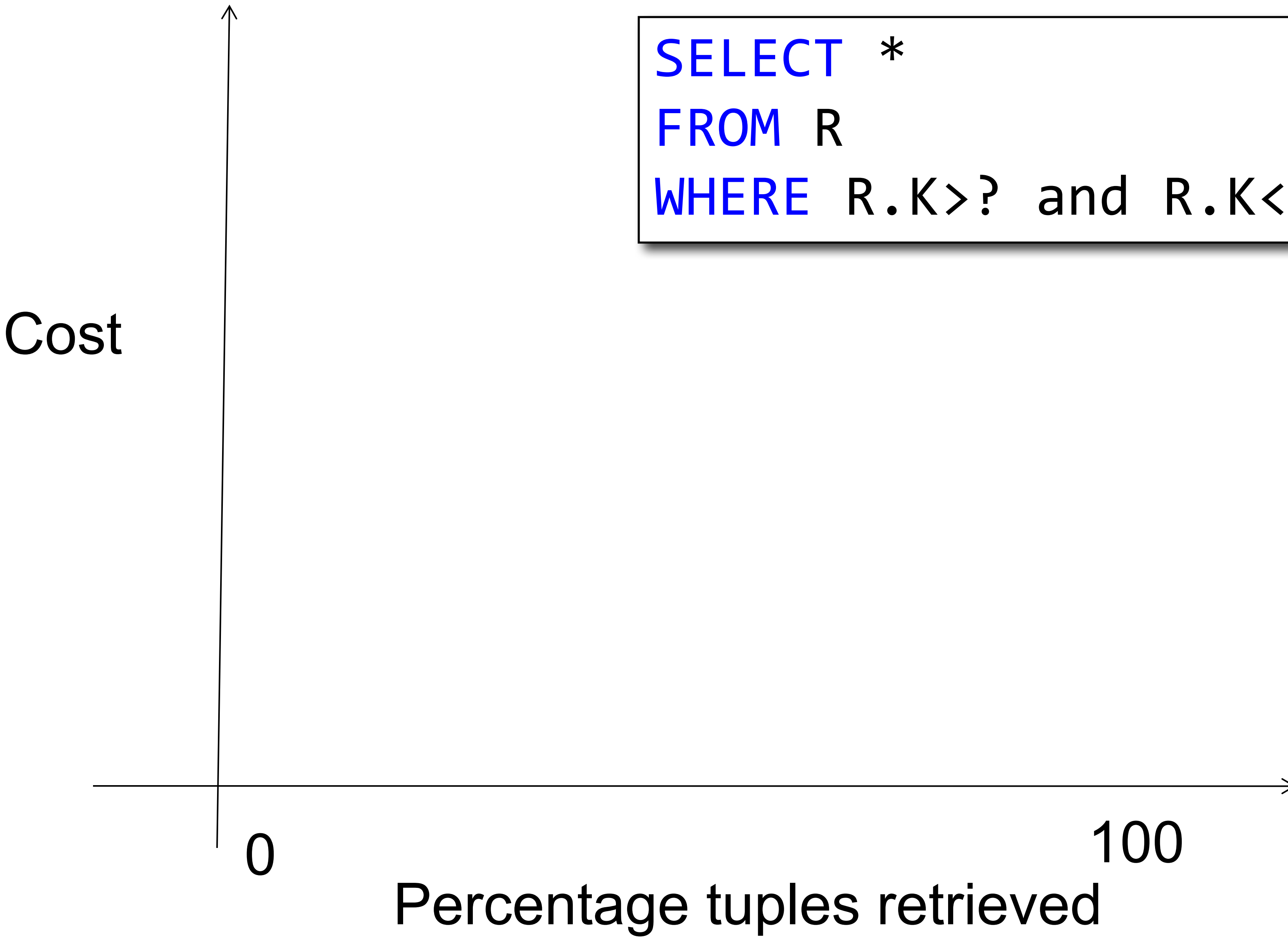
Try to choose indexes that speed-up multiple queries

TO CLUSTER OR NOT

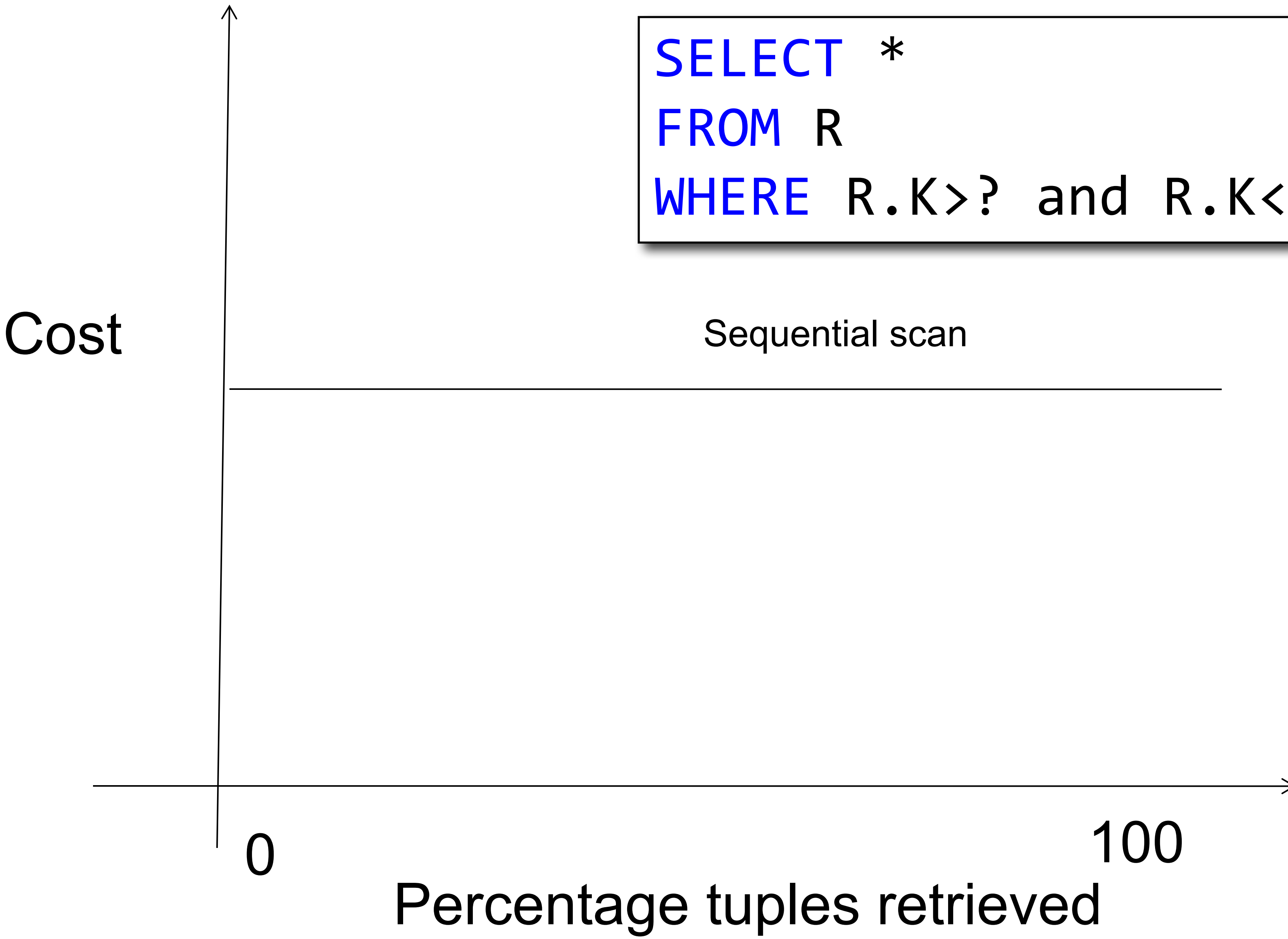
Range queries benefit mostly from clustering

Point indexes do *not* need to be clustered: they work equally well unclustered

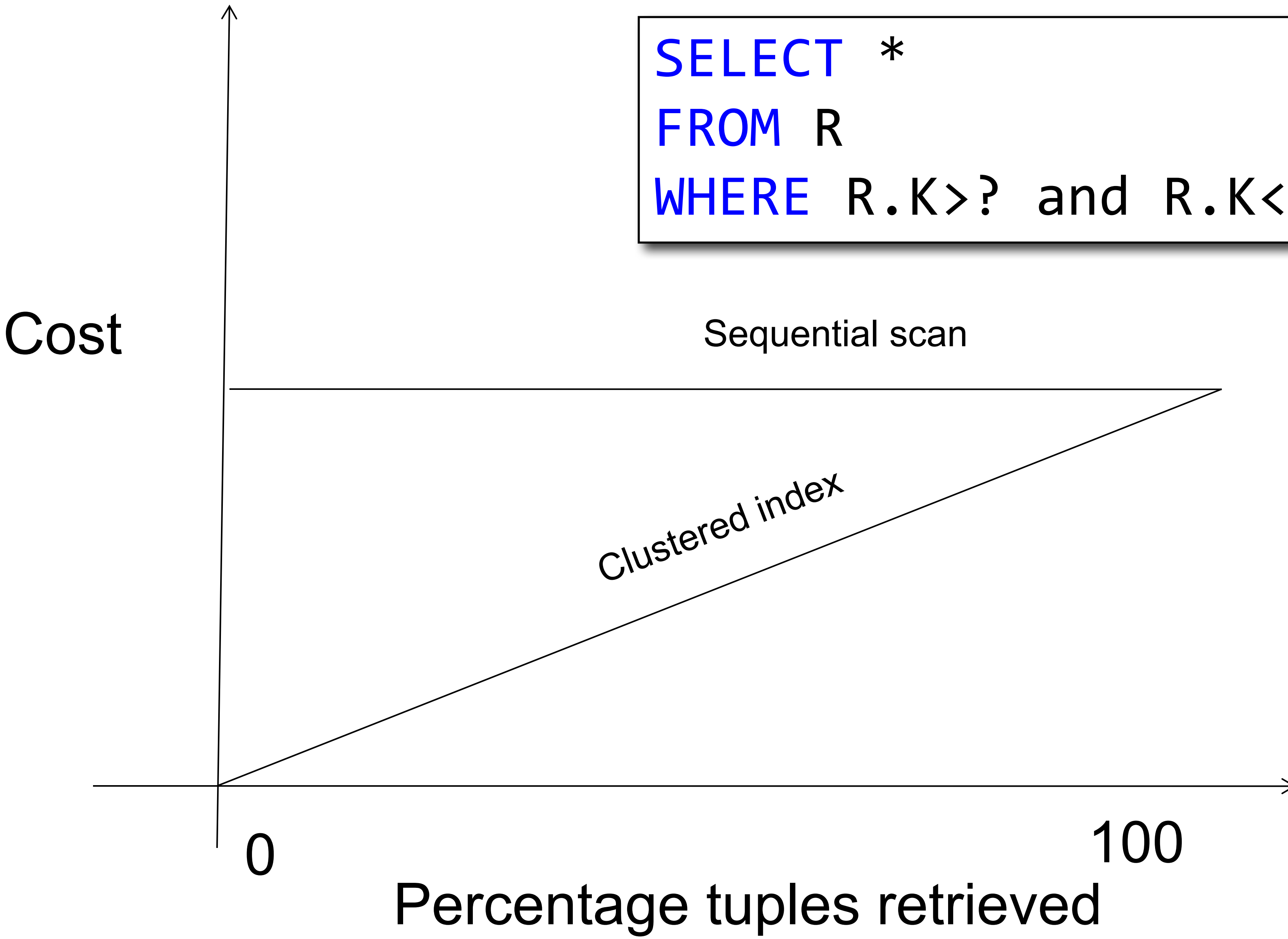
```sql
SELECT *
FROM R
WHERE R.K>? and R.K<?
```



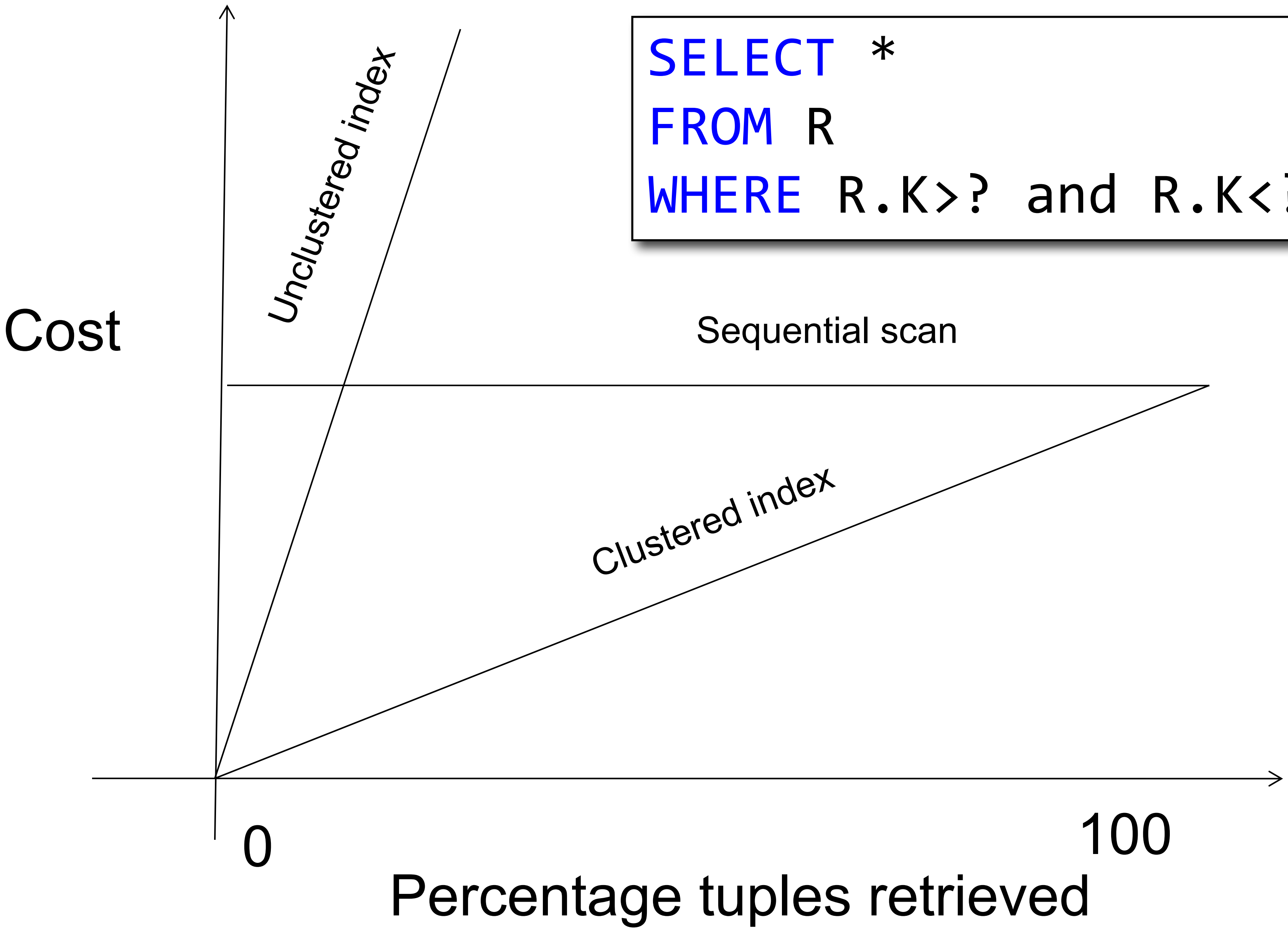
```
SELECT *
FROM R
WHERE R.K>? and R.K<?
```



```sql
SELECT *
FROM R
WHERE R.K>? and R.K<?
```



```
SELECT *
FROM R
WHERE R.K>? and R.K<?
```



CHOOSING INDEX IS NOT ENOUGH

To estimate the cost of a query plan, we still need to consider other factors:

- How each operator is implemented
- The cost of each operator
- Let’s start with the basics

COST PARAMETERS

Cost = I/O + CPU + Network BW

- We will focus on I/O in this class

Parameters (a.k.a. statistics):

- **B(R)** = # of blocks (i.e., pages) for relation R
- **T(R)** = # of tuples in relation R
- **V(R, a)** = # of distinct values of attribute a

COST PARAMETERS

Cost = I/O + CPU + Network BW

- We will focus on I/O in this class

Parameters (a.k.a. statistics):

- **B(R)** = # of blocks (i.e., pages) for relation R
- **T(R)** = # of tuples in relation R
- **V(R, a)** = # of distinct values of attribute a

When **a** is a key, **V(R,a) = T(R)**
When **a** is not a key, **V(R,a)** can be anything <= **T(R)**

COST PARAMETERS

Cost = I/O + CPU + Network BW

- We will focus on I/O in this class

Parameters (a.k.a. statistics):

- **B(R)** = # of blocks (i.e., pages) for relation R
- **T(R)** = # of tuples in relation R
- **V(R, a)** = # of distinct values of attribute a

When **a** is a key, **V(R,a) = T(R)**
When **a** is not a key, **V(R,a)** can be anything <= **T(R)**

DBMS collects statistics about base tables must infer them for intermediate results

SELECTIVITY FACTORS FOR CONDITIONS

A = c /* $\sigma_{A=c}(R)$ */

- Selectivity = 1/V(R,A)

A < c /* $\sigma_{A<c}(R)$*/

- Selectivity = (c - min(R, A))/(max(R,A) - min(R,A))

c1 < A < c2 /* $\sigma_{c1<A<c2}(R)$*/

- Selectivity = (c2 – c1)/(max(R,A) - min(R,A))

COST OF READING DATA FROM DISK

Sequential scan for relation R costs B(R)

Index-based selection

- Estimate selectivity factor **f** (see previous slide)
- Clustered index: f***B(R)**
- Unclustered index f***T(R)**

Note: we ignore I/O cost for index pages

INDEX BASED SELECTION

Example:

$B(R) = 2000$
$T(R) = 100{,}000$
$V(R, a) = 20$

cost of $\sigma_{a=v}(R) = ?$

Table scan:

Index based selection:

INDEX BASED SELECTION

Example:

$B(R) = 2000$
$T(R) = 100{,}000$
$V(R, a) = 20$

cost of $\sigma_{a=v}(R)$ = ?

Table scan: B(R) = 2,000 I/Os

Index based selection:

INDEX BASED SELECTION

Example:

$B(R) = 2000$
$T(R) = 100{,}000$
$V(R, a) = 20$

cost of $\sigma_{a=v}(R) = ?$

Table scan: $B(R) = 2{,}000$ **I/Os**

Index based selection:

- If index is clustered:
- If index is unclustered:

INDEX BASED SELECTION

Example:

$B(R) = 2000$
$T(R) = 100{,}000$
$V(R, a) = 20$

cost of $\sigma_{a=v}(R)$ = ?

Table scan: $B(R)$ = 2,000 I/Os

Index based selection:

- If index is clustered: $B(R) * 1/V(R,a) = 100$ I/Os
- If index is unclustered:

INDEX BASED SELECTION

Example:

$B(R) = 2000$
$T(R) = 100{,}000$
$V(R, a) = 20$

cost of $\sigma_{a=v}(R)$ = ?

Table scan: B(R) = 2,000 I/Os

Index based selection:

- If index is clustered: B(R) * 1/V(R,a) = 100 I/Os
- If index is unclustered: T(R) * 1/V(R,a) = 5,000 I/Os

INDEX BASED SELECTION

Example:

$B(R) = 2000$
$T(R) = 100{,}000$
$V(R, a) = 20$

cost of $\sigma_{a=v}(R)$ = ?

Table scan: $B(R)$ = 2,000 I/Os

Index based selection:

- If index is clustered: $B(R) * 1/V(R,a)$ = 100 I/Os
- If index is unclustered: $T(R) * 1/V(R,a)$ = 5,000 I/Os

Lesson: Don't build unclustered indexes when $V(R,a)$ is small !

OUTLINE

Join operator algorithms

- One-pass algorithms (Sec. 15.2 and 15.3)
- Index-based algorithms (Sec 15.6)

Note about readings:

- In class, we discuss only algorithms for joins
- Other operators are easier: read the book

JOIN ALGORITHMS

Hash join

Nested loop join

Sort-merge join

HASH JOIN

Hash join: R $\bowtie$ S

Scan R, build buckets in main memory

Then scan S and join

Cost: B(R) + B(S)

Which relation to build the hash table on?

HASH JOIN

Hash join: R $\bowtie$ S

Scan R, build buckets in main memory

Then scan S and join

Cost: B(R) + B(S)

Which relation to build the hash table on?

One-pass algorithm when B(R) ≤ M

- M = number of memory pages available

HASH JOIN EXAMPLE

Patient(pid, name, address)

Insurance(pid, provider, policy_nb)

Patient $\bowtie$ Insurance

Two tuples per page

Patient

1	'Bob'	'Seattle'
2	'Ela'	'Everett'

3	'Jill'	'Kent'
4	'Joe'	'Seattle'

Insurance

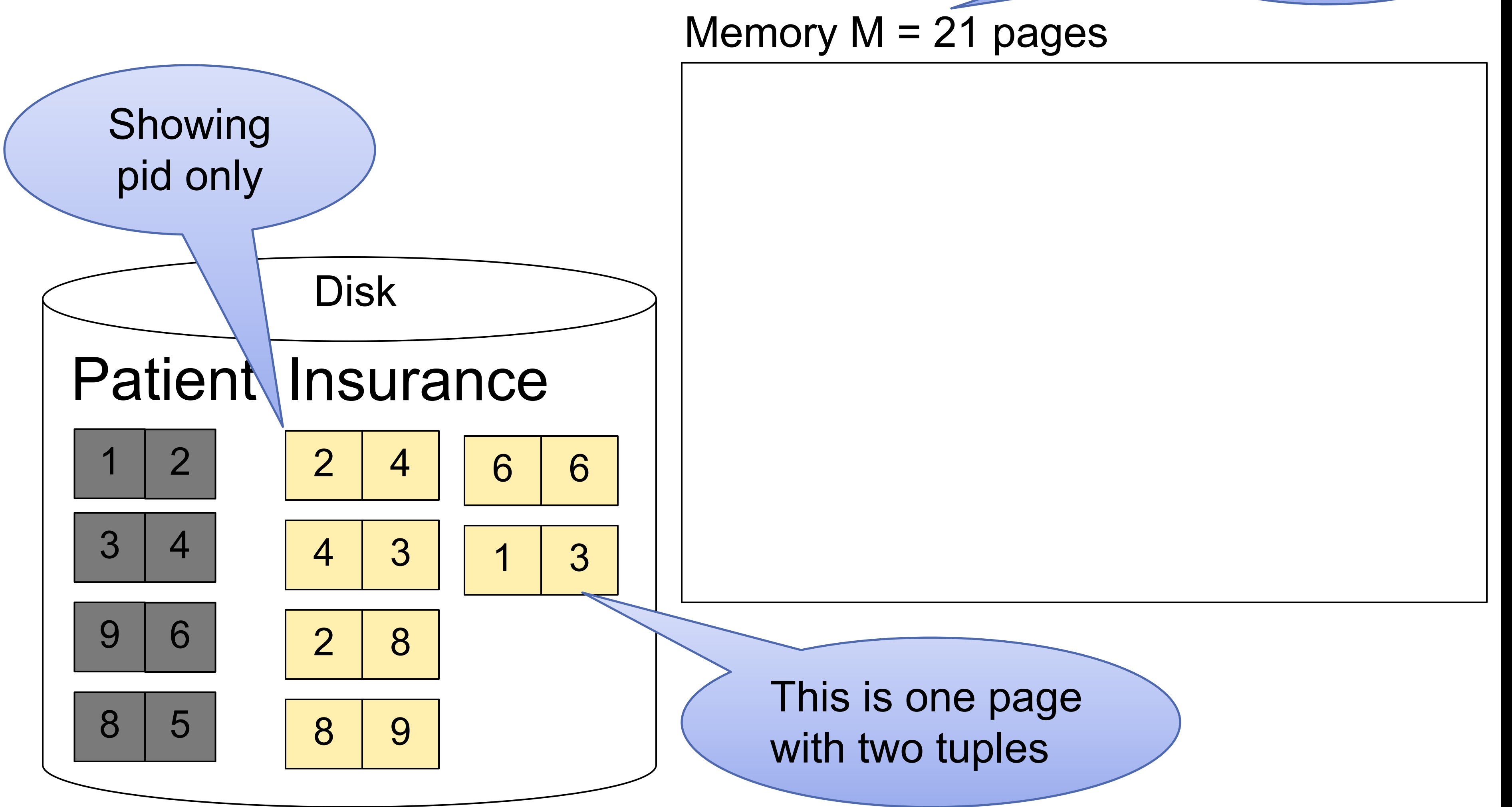
2	'Blue'	123
4	'Prem'	432

4	'Prem'	343
3	'GrpH'	554

HASH JOIN EXAMPLE

Patient $\bowtie$ Insurance

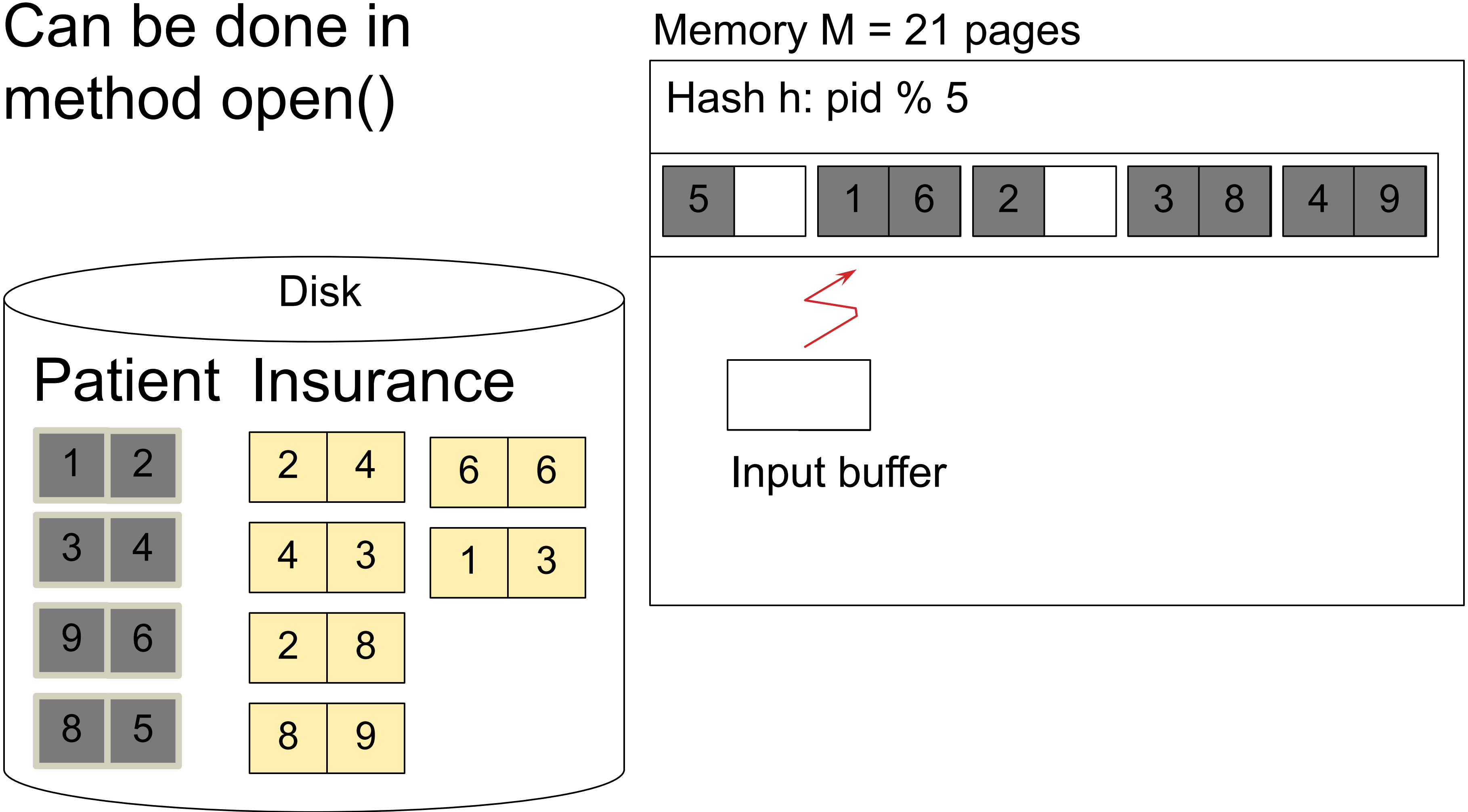
Some large-enough #



HASH JOIN EXAMPLE

Step 1: Scan Patient and build hash table in memory

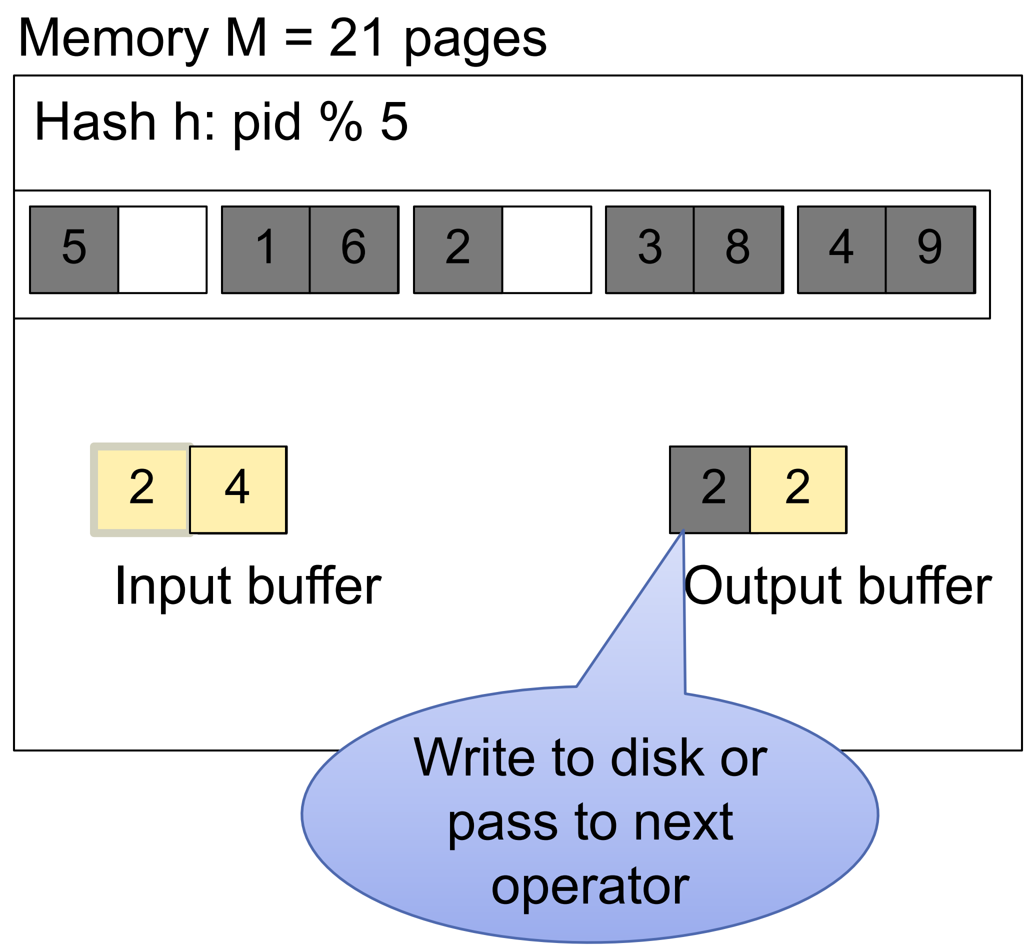
Can be done in method open()

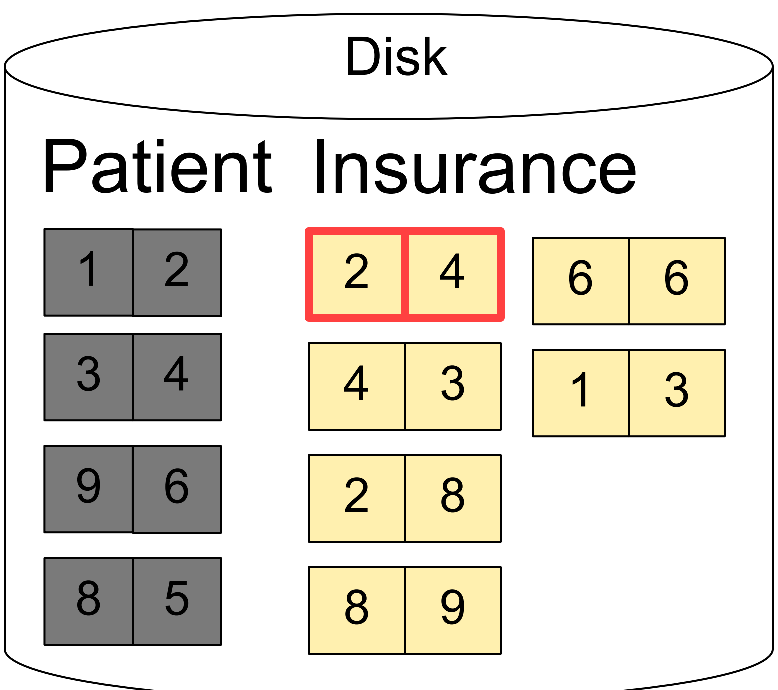


HASH JOIN EXAMPLE

Step 2: Scan Insurance and probe into hash table

Done during calls to next()

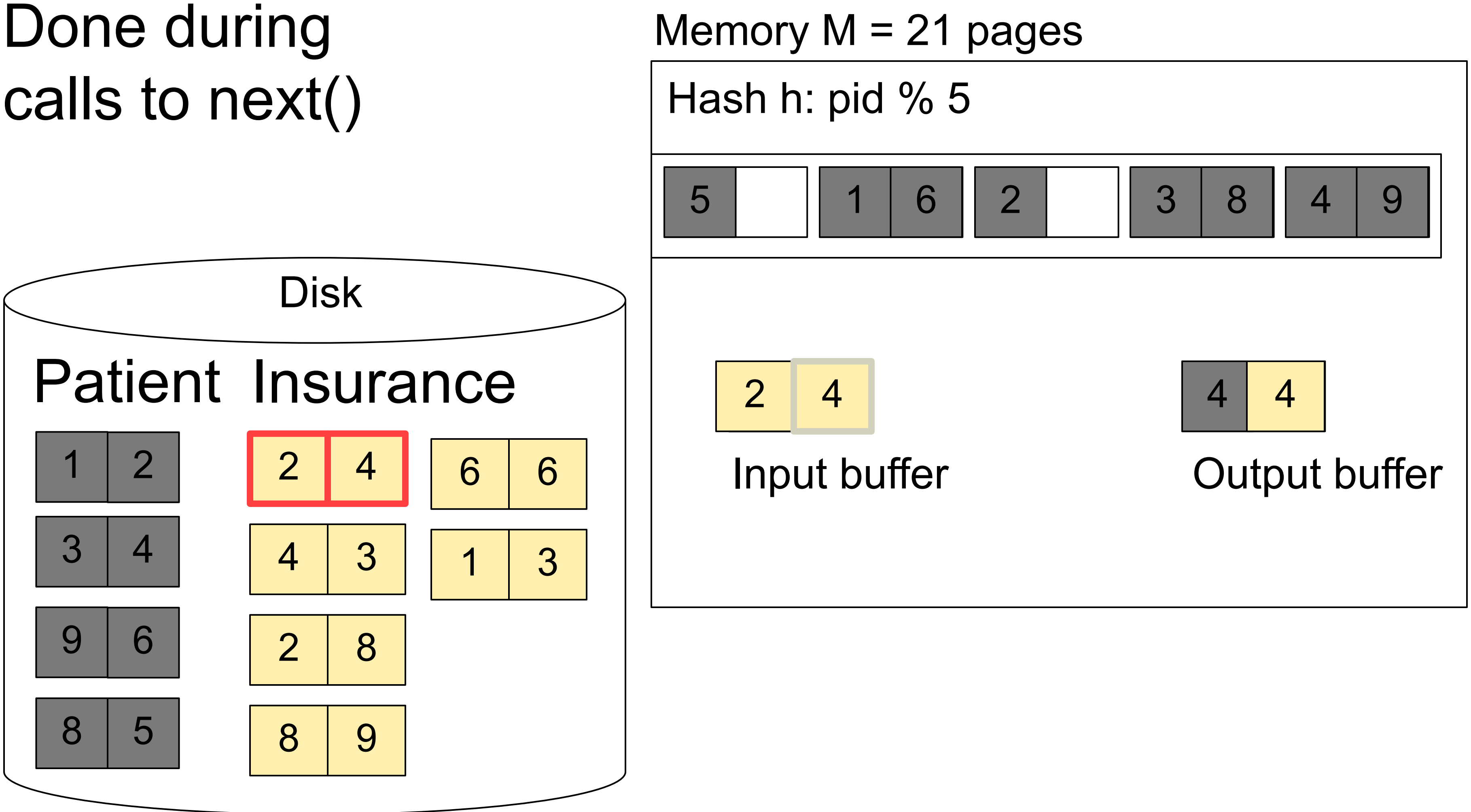




HASH JOIN EXAMPLE

Step 2: Scan Insurance and probe into hash table

Done during calls to next()

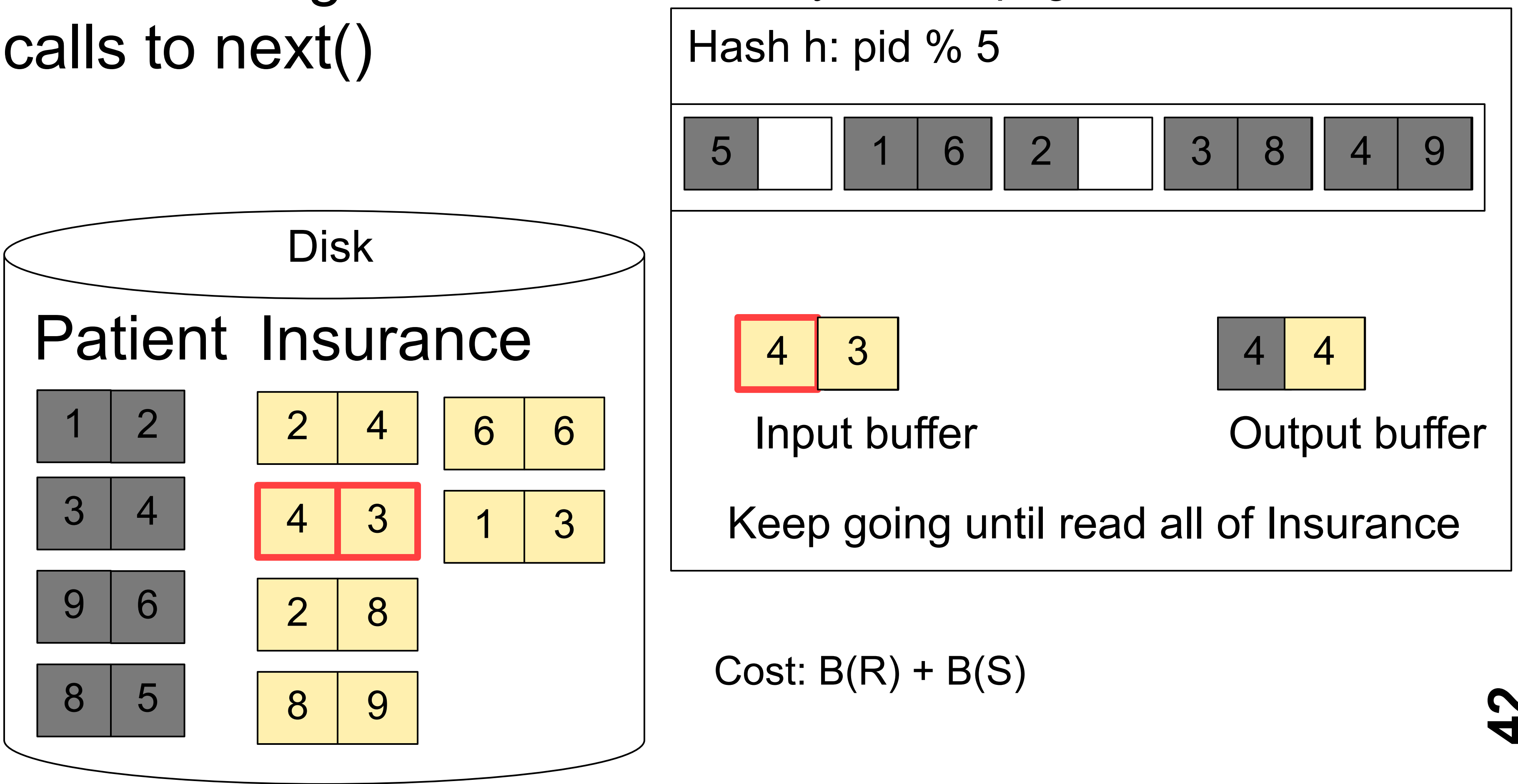


HASH JOIN EXAMPLE

Step 2: Scan Insurance and probe into hash table

Done during calls to next()

Memory M = 21 pages



Cost: B(R) + B(S)

NESTED LOOP JOINS

Tuple-based nested loop R ⋈ S

R is the outer relation, S is the inner relation

```
for each tuple t1 in R do
   for each tuple t2 in S do
      if t1 and t2 join then output (t1,t2)
```

What is the Cost?

NESTED LOOP JOINS

Tuple-based nested loop $R \bowtie S$

R is the outer relation, S is the inner relation

```
for each tuple t1 in R do
   for each tuple t2 in S do
      if t1 and t2 join then output (t1,t2)
```

Cost: $B(R) + T(R)\,B(S)$

Multiple-pass since S is read many times

What is the Cost?

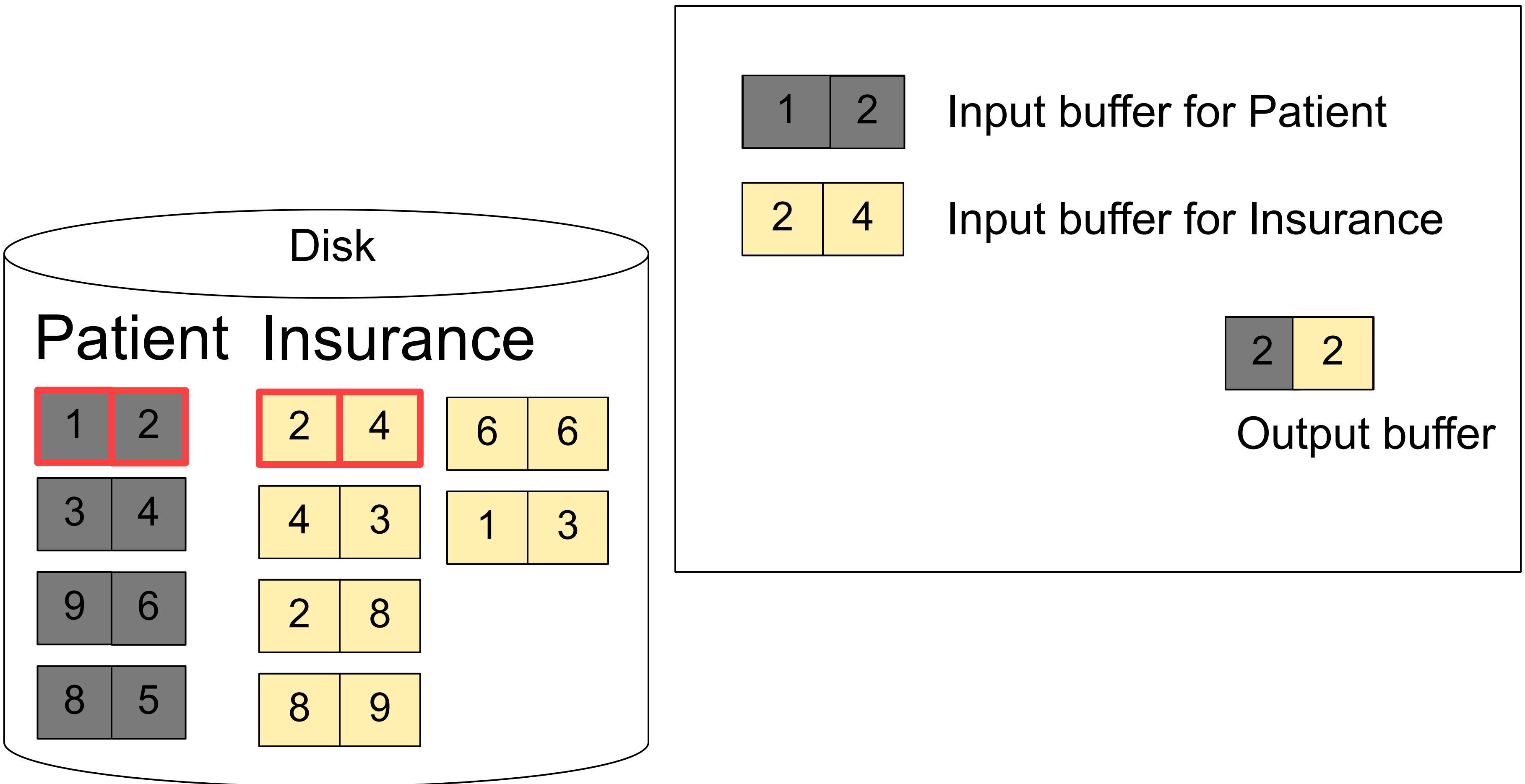
PAGE-AT-A-TIME REFINEMENT

```
for each page of tuples r in R do
  for each page of tuples s in S do
    for all pairs of tuples t1 in r, t2 in s
      if t1 and t2 join then output (t1,t2)
```

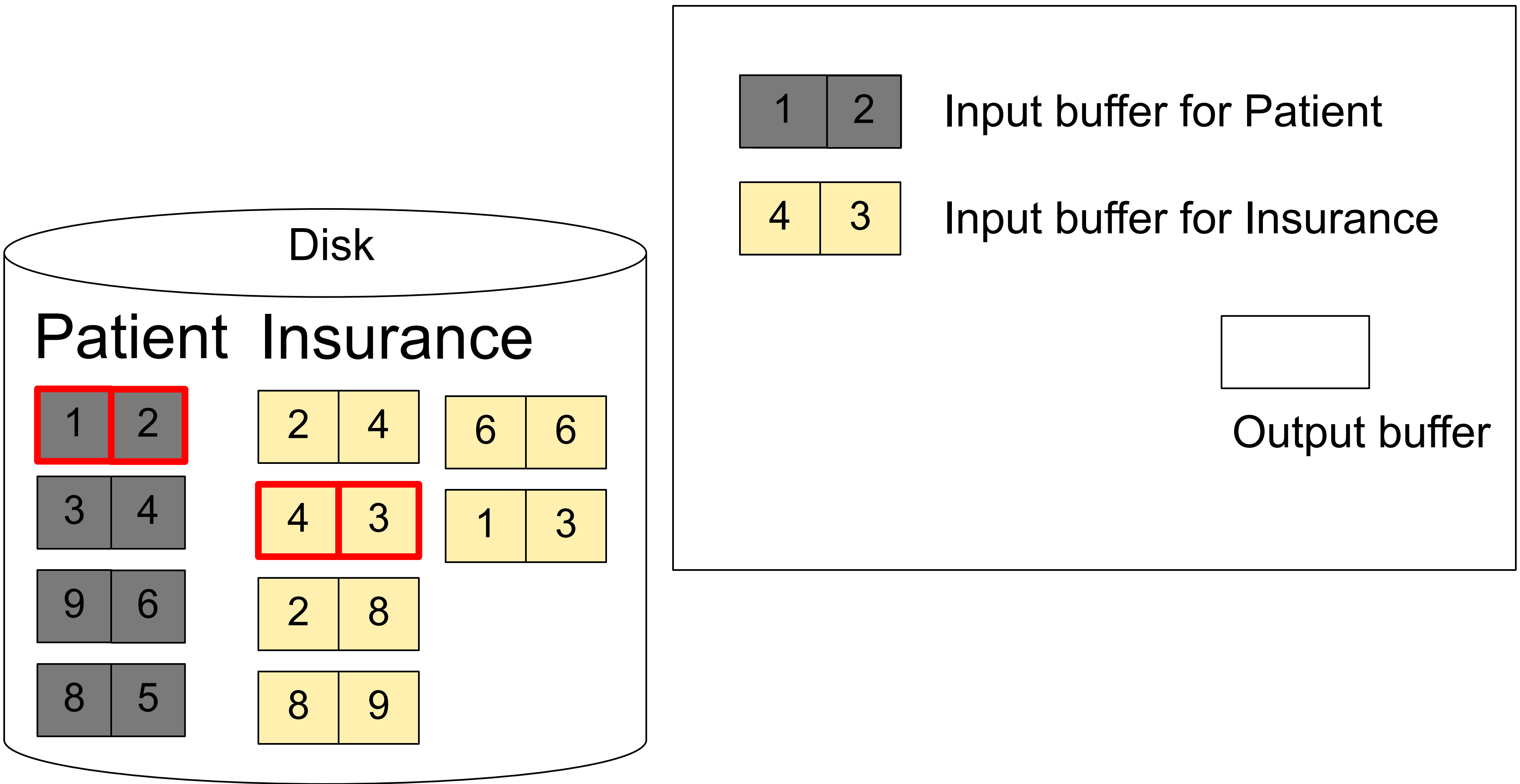
Cost: B(R) + B(R)B(S)

What is the Cost?

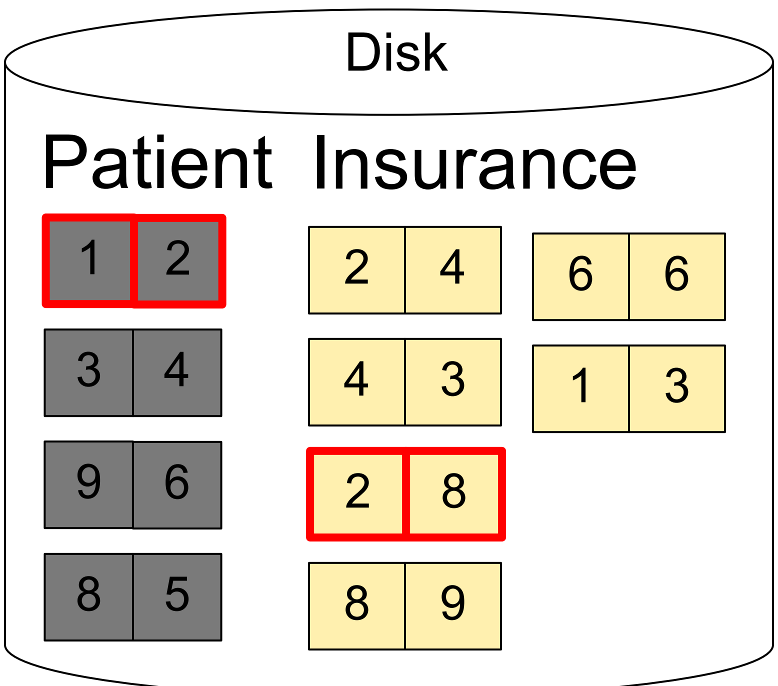
PAGE-AT-A-TIME REFINEMENT

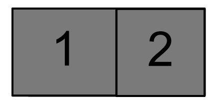


PAGE-AT-A-TIME REFINEMENT

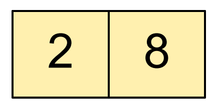


PAGE-AT-A-TIME REFINEMENT



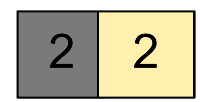


Input buffer for Patient



Input buffer for Insurance

Keep going until read all of Insurance



Output buffer

Then repeat for next page of Patient… until end of Patient

Cost: B(R) + B(R)B(S)

BLOCK-NESTED-LOOP REFINEMENT

```
for each group of M-1 pages r in R do
  for each page of tuples s in S do
     for all pairs of tuples t1 in r, t2 in s
          if t1 and t2 join then output (t1,t2)
```

Cost: B(R) + B(R)B(S)/(M-1)

What is the Cost?