Database Systems
CSE 414

Section 10: Big Data & Review

Non-Parallel Query Evaluation

Example Schema

`Product(pid, name, category)`

- 10,000 tuples and 1,000 blocks
- 40 different categories

`Order(store, pid, price, quantity)`

- 1,000,000 tuples and 50,000 blocks
- prices range from $1 to $100

Example Query

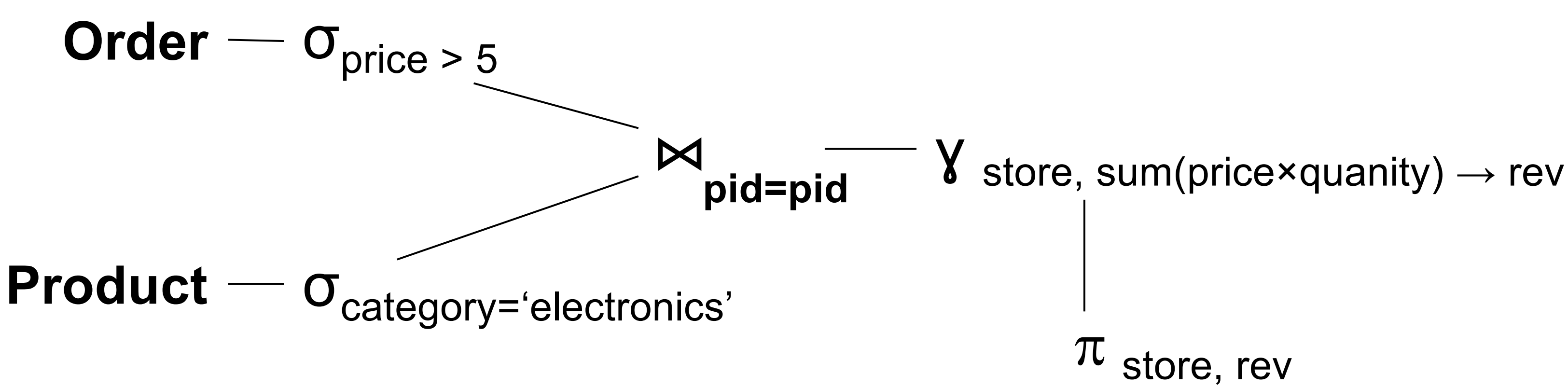
Compute the total revenue, for each store, from electronics costing more than $5 each:

```
SELECT o.store, sum(o.price * o.quantity)
FROM Order o, Product p
WHERE o.pid = p.pid AND o.price > 5 AND
    p.category = 'electronics'
GROUP BY o.store
```

Problem 1

Give an RA expression that:

- computes the result of the query
- **does not** benefit from the indexes already present



Problem 2

Estimate the cost of the RA expression from Problem 1 after filling in physical implementation details

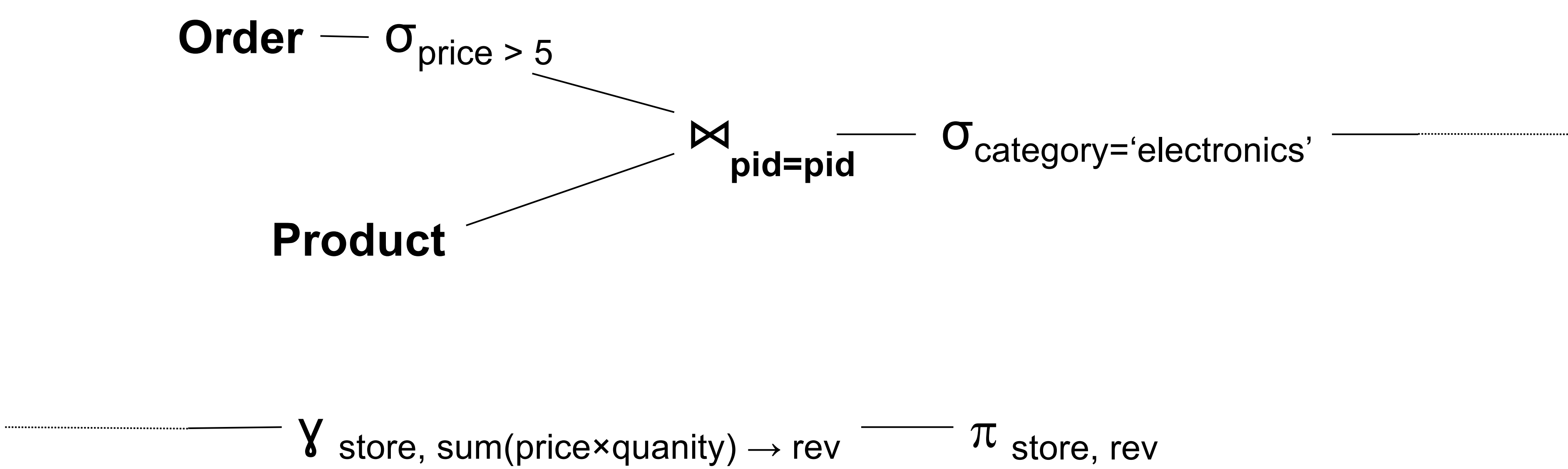
- assume grouping / aggregation can be done on the fly

- Details:
  - nested loop join
  - write Products to temp T1
  - grouping / aggregation done with in memory hash table
- Scan Product & writing to T1 costs 50k + 1k + 25
- Nested loop join costs 47.5k * 25 = 1,125k
- Total cost is 1,238,525 blocks (~1M is fine)

Problem 3

Give an RA expression that:

- computes the result of the query
- **does** benefit from the indexes already present



Problem 4

Estimate the cost of the RA expression from Problem 3 after filling in physical implementation details

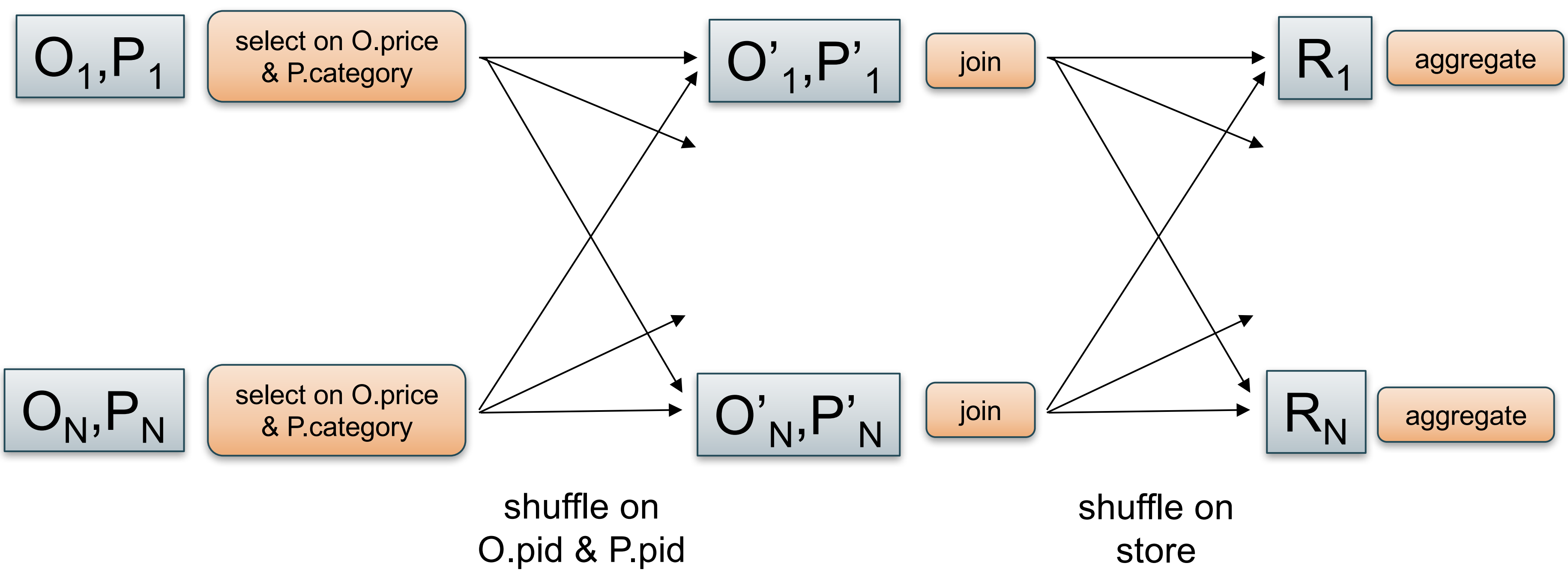
– assume grouping / aggregation can be done on the fly

- Details:
  - nested loop join using index on Product(pid)
  - grouping / aggregation done with in memory hash table
- Lookup of Product costs 1 block
- Nested loop join costs 50k + 950k * 1 = 1000k
- Total cost is ~1M blocks (everything else on the fly)

Parallel Query Evaluation

Problem 5

Draw a pipeline that computes the same result in a parallel fashion using N nodes



Problem 6

Estimate the cost of executing the pipeline of Problem 5

- Only costs are on disk reads of input
  - (everything should fit in memory)
- Each worker reads 50k/N + 1k/N blocks
- Since all workers are reading simultaneous, wait time is time to read 51k/N blocks (plus lower order work)

Problem 7

1. Does your analysis predict a linear speedup as more nodes are added?

    Yes

2. Does your analysis predict a linear scaleup as more nodes are added?

    Yes

3. How realistic is this?

    Fair with a small number of machines, but expect stragglers to be noticeable with 1000s

Problem 8

Describe how to achieve a similar speedup with MapReduce

- MapReduce does only one shuffle, so we need 2 jobs
- First job:
  - map Orders to (pid,('O', …)) and Products to (pid,('P',...) for those rows that satisfy selection criteria
  - reducer adds product info to each order in the list
    - note: only one Product in each list since pid is primary key
- Second job:
  - map Order+Product to (store, (...))
  - reducer sums revenue and outputs (store, revenue)

Problem 9

Would your MapReduce have the same IO cost and speedup as the pipeline from problem 6?

- MapReduce writes intermediate results to disk resulting in more IO
  - Two intermediate results and two outputs written
  - None of these are larger than the input, though, so the total cost is no more than 7x the ideal pipeline
    - really 6x since the final output is small
- Despite a constant factor more IO, it should still have a linear speedup (in principle).