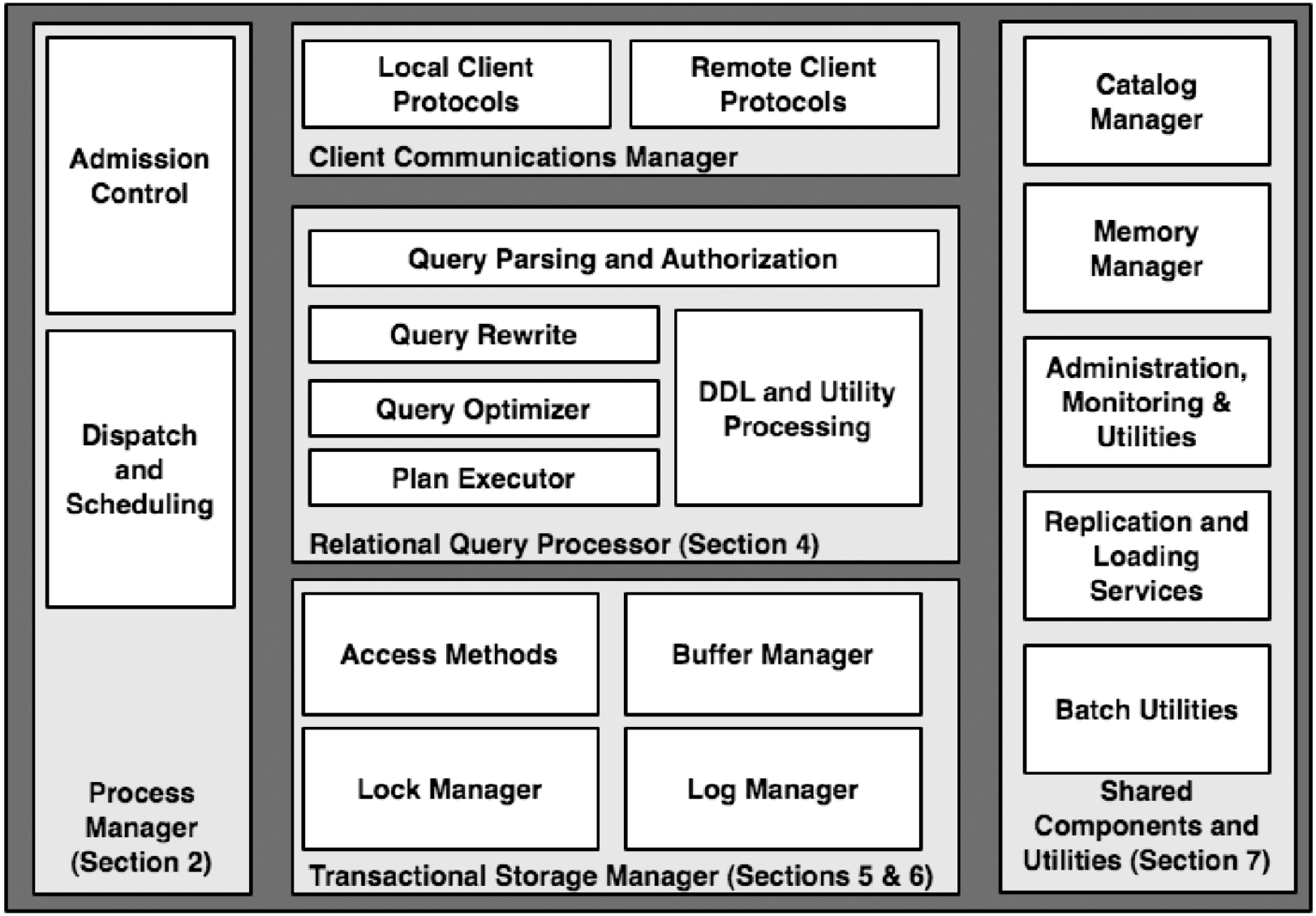
CS5226 Lecture 1
Introduction

Database Tuning

- Make a database application run more quickly
  - higher throughput
  - lower response time
- Auto-tuning / self-tuning
  - Better performance
  - Easier manageability
- Query workload
  - Online transaction processing (OLTP)
  - Decision support systems (DSS) / Online analytical processing (OLAP) / Data warehousing

Anatomy of DBMS

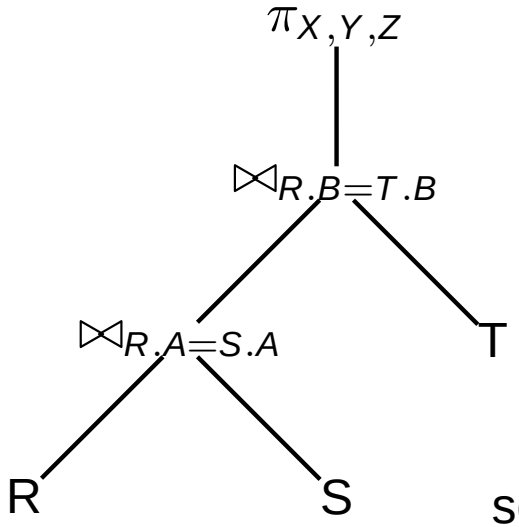


(Hellerstein, Stonebraker, Hamilton, 2007)

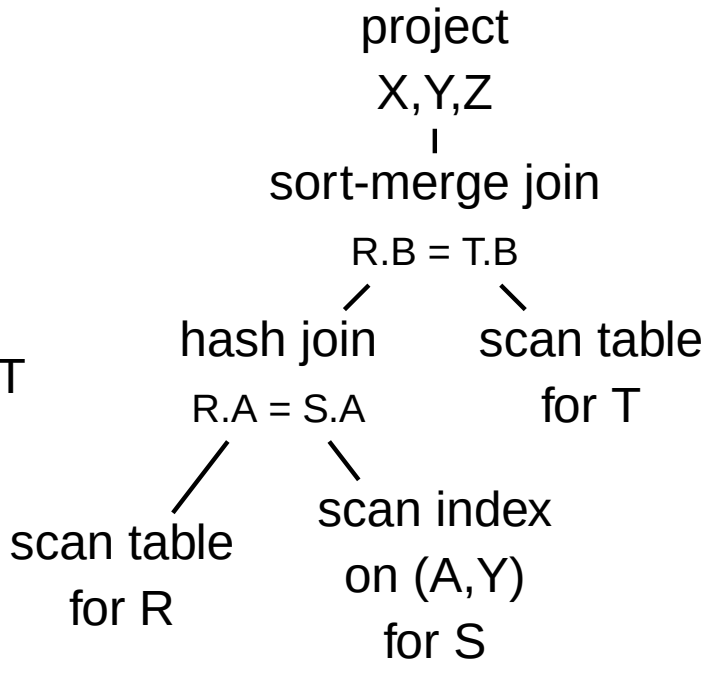
Query Optimization

select R.X, S.Y, T.Z
from R, S, T
where R.A = S.A
and R.B = T.B

Query



Internal Query Representation



Physical Query Plan

Performance Tuning Knobs

- Schema tuning
- Query tuning
- Index & materialized view selection
- Statistics tuning
- Concurrency control tuning
- Data partitioning
- Memory tuning
- Hardware tuning

Schema Tuning

CourseInfo

Module	Prof	Room	Building	Time
CS101	Turing	LT 1	CS	0800
CS400	Turing	LT 1	CS	1400
MU300	Bach	LT 2	Math	1400
MA200	Newton	LT 2	Math	1000
CS101	Turing	LT 2	Math	1200

Facility

Room	Building
LT 1	CS
LT 2	Math

Course

Module	Prof
CS101	Turing
CS400	Turing
MU300	Bach
MA200	Newton

Schedule

Room	Time	Module
LT 1	0800	CS101
LT 1	1400	CS400
LT 2	1400	MU200
LT 2	1000	MA200
LT 2	1200	CS101

Query Tuning

Q1:
```
select   c.cname
from     Customer c
where    1000 < (select sum(o.totalprice)
                 from   Order o
                 where  o.cust# = c.cust#)
```

Q2:
```
select   c.cname
from     Customer c join Order o
         on c.cust# = o.cust#
group by c.cust#, c.cname
having   1000 < sum(o.totalprice)
```

Query Tuning (cont.)

Q3:
```
select    c.cust#, c.cname, sum(o.totalprice) as T
from      Customer c join Order o
          on c.cust# = o.cust#
group by  c.cust#, cname
```

Q4:
```
select    c.cust#, cname, T
from      customer c,
          (select   cust#, sum(totalprice) as T
          from      Order
          group by  cust#) as o
where     c.cust# = o.cust#
```

Query Tuning (cont.)

Q5: **select** **distinct** R.A, S.X
from R, S
where R.B = S.Y

Q6: **select** R.A, S.X
from R, S
where R.B = S.Y

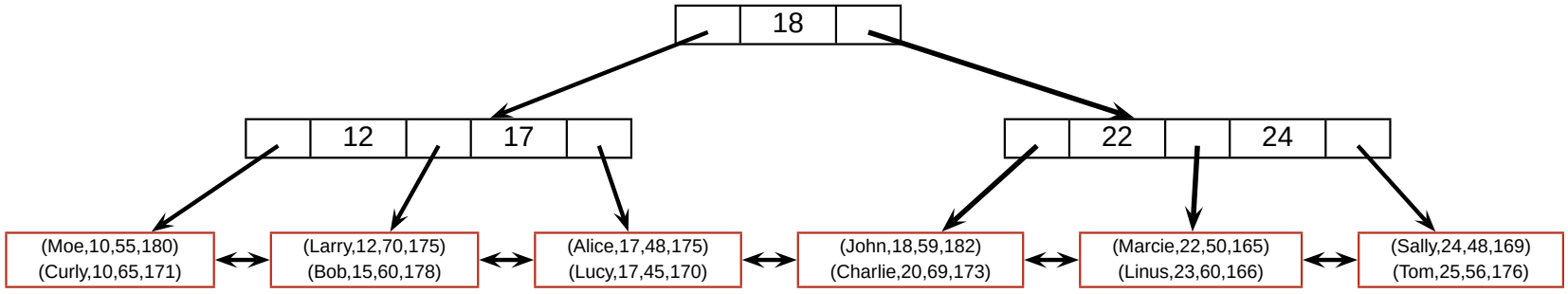
Index Tuning

select A, B, C
from R
where $10 < A < 20$

Access methods for selection queries:

- Table scan
- Use one or more indexes

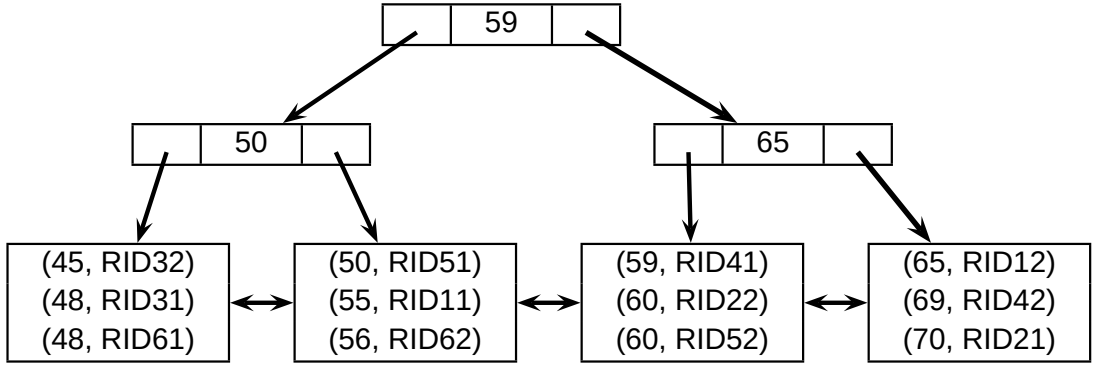
B$^+$-tree index



Clustered index on (age)

Relation R

name	age	weight	height
Moe	10	55	180
Curly	10	65	171
Larry	12	70	175
Bob	15	60	178
Alice	17	48	175
Lucy	17	45	170
John	18	59	182
Charlie	20	69	173
Marcie	22	50	165
Linus	23	60	166
Sally	24	48	169
Tom	25	56	176



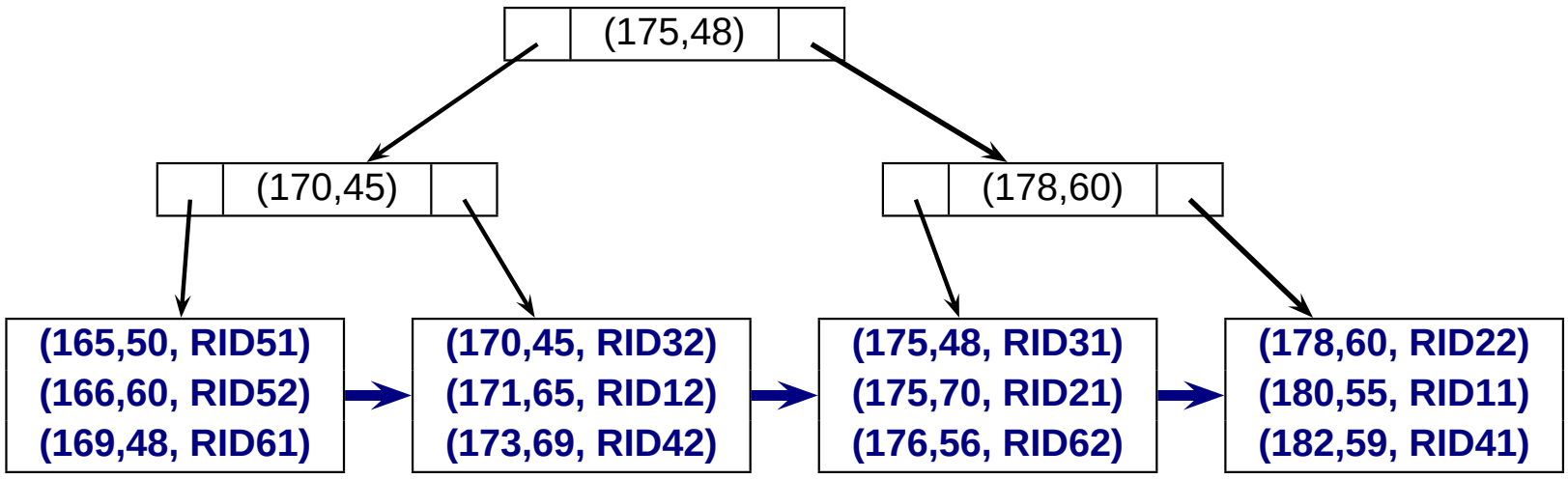
Unclustered index on (weight)

Index access methods

- Index scan
- Index seek [+ RID lookup]
- Index intersection [+ RID lookup]

Index scan

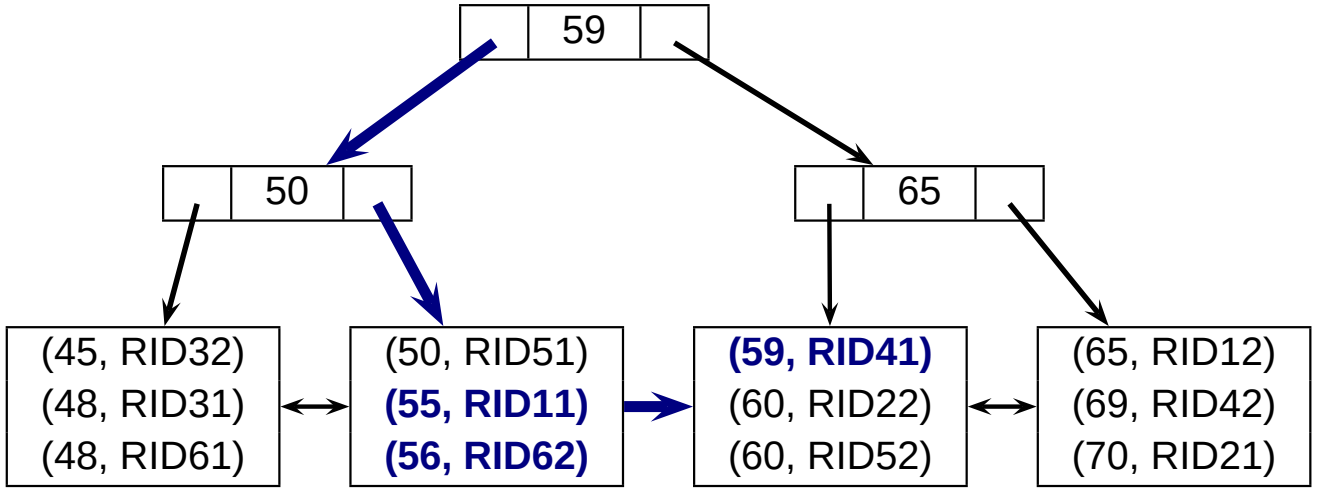
select height
from Student



Index on (height,weight)

Index seek

select weight
from Student
where weight **between** 55 **and** 65

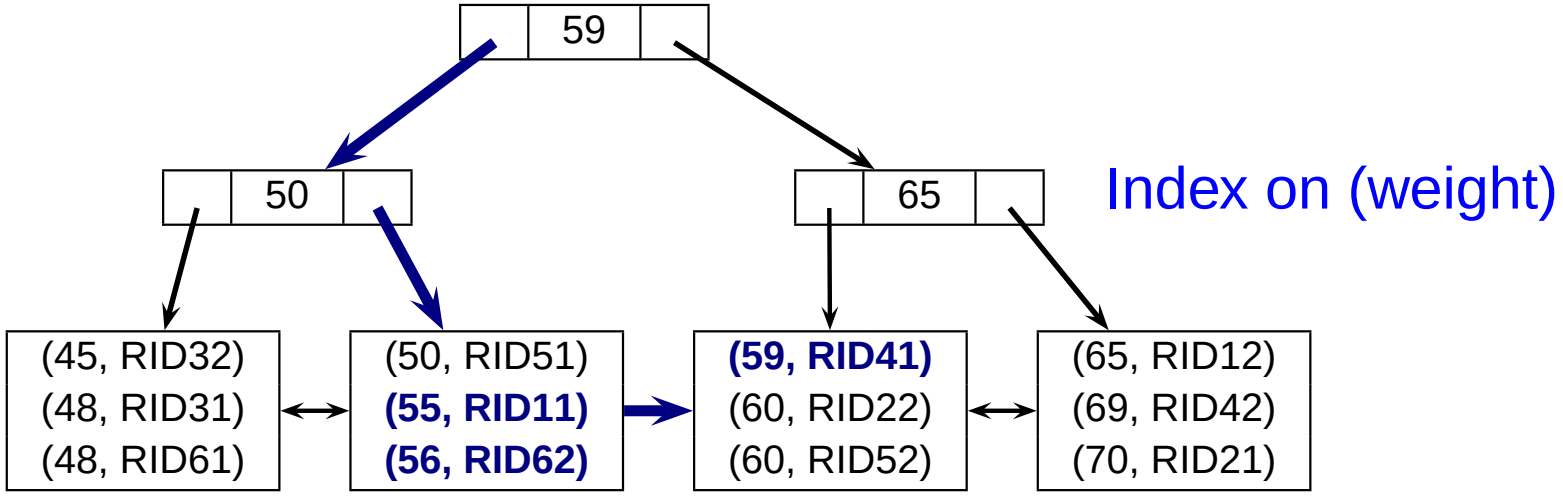


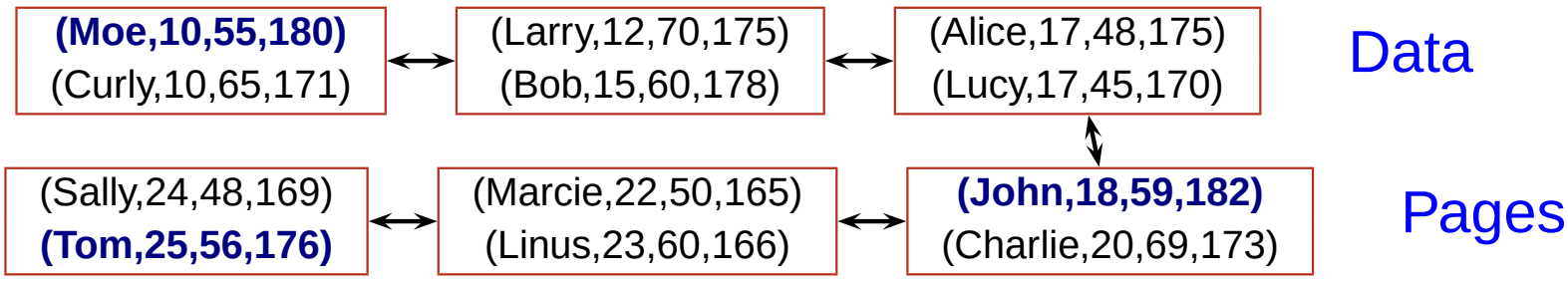
Index on (weight)

Index seek + RID lookups

select name
from Student
where weight **between** 55 **and** 59

select weight
from Student
where weight **between** 55 **and** 59
and age $\geq$ 20

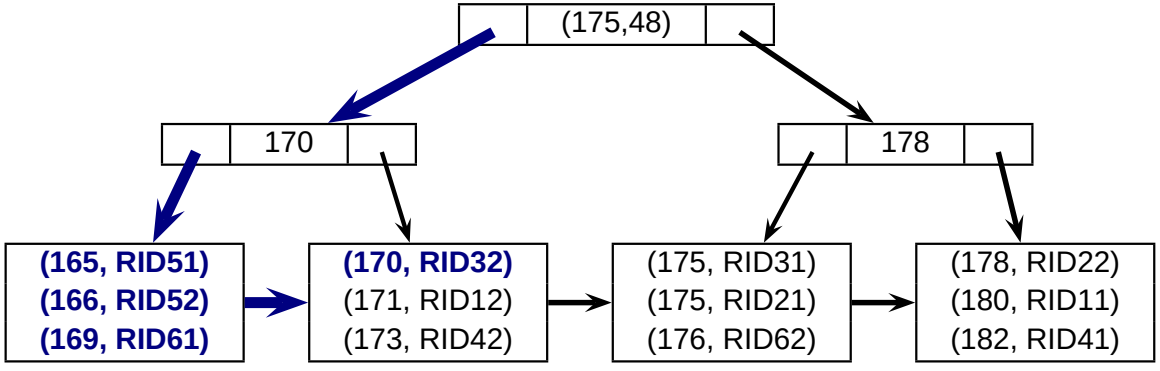




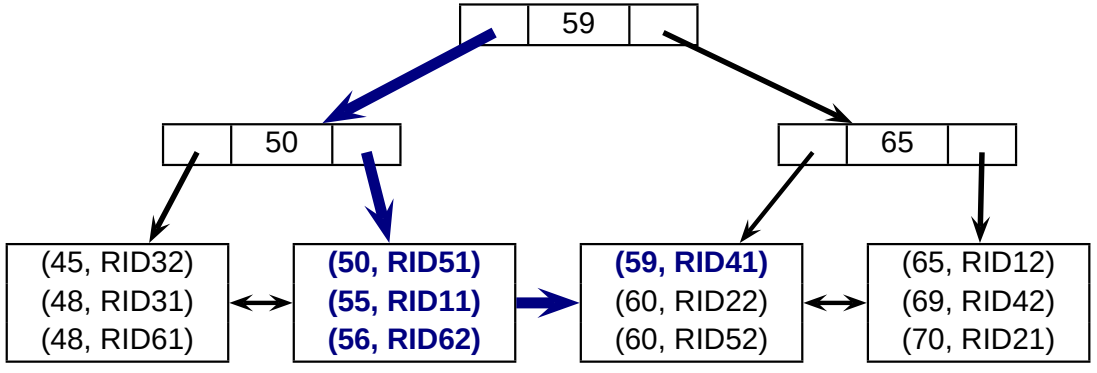
Index intersection

select height, weight **from** Student
where height **between** 164 **and** 170
and weight **between** 50 **and** 59

Index on (height)



Index on (weight)



Index Tuning

Q1: **select** A, B, C
from R
where $10 < A < 20$
and $20 < B < 100$

Q2: **select** B, C, D
from R
where $50 < B < 100$
and $60 < D < 80$

Materialized View Tuning

Q1: **select** R.B
from R, S
where R.A = S.X
and S.Y > 100

MV1: **select** R.A, R.B, S.X, S.Y
from R, S
where R.A = S.X

Q1': **select** B
from MV1
where Y > 100

Tuning of indexes & materialized views

Given a query workload and a disk space constraint, what is the optimal configuration of indexes & materialized views to optimize the performance of the workload?

Tuning of statistics

Examples of statistics:

- table cardinality
- statistics for each column:
  - number of distinct values
  - highest & lowest values
  - frequent values
  - data distribution statistics
- multi-column statistics

Issues

- What statistics to collect?
- When to collect/refresh statistics?

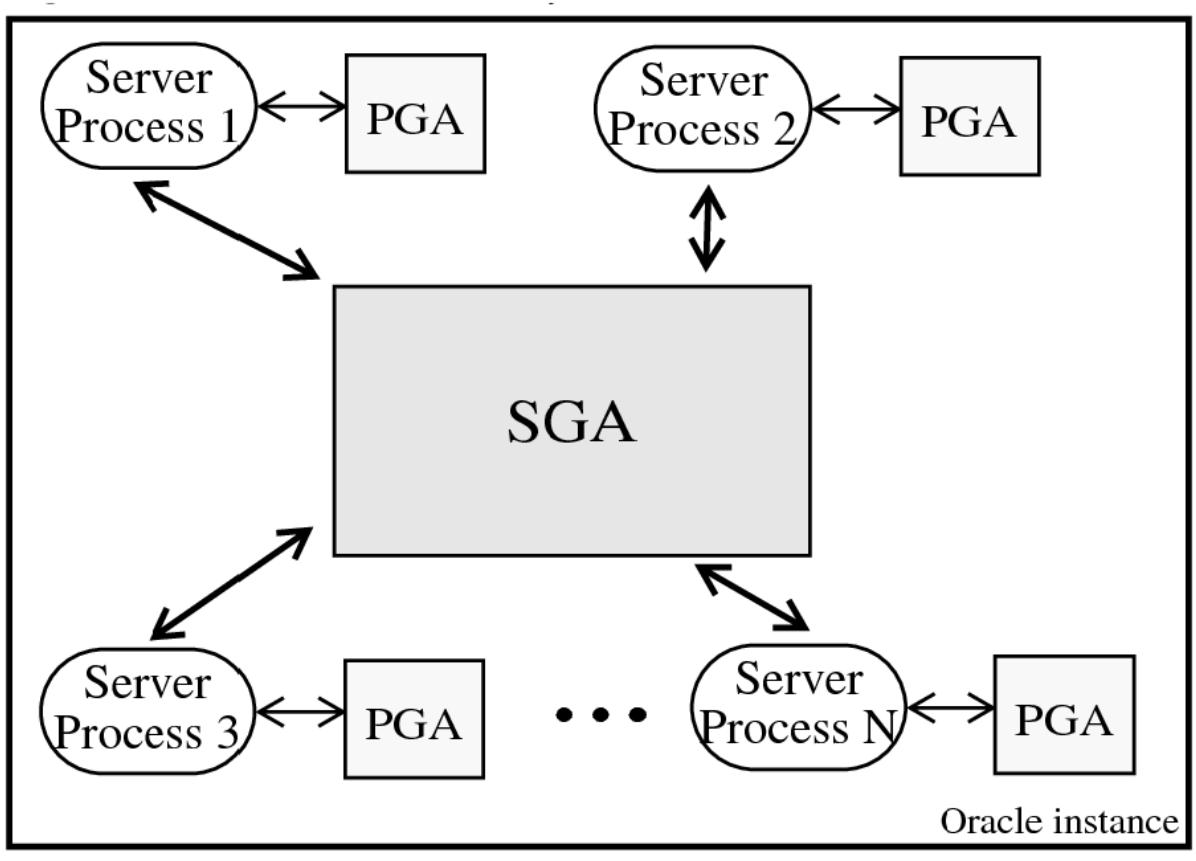
Tuning of concurrency control

- Concurrency control protocols
  - Two-phase locking
  - Snapshot isolation
- Consistency vs concurrency tradeoff
- ANSI SQL isolation levels

Isolation Level	Dirty Read	Unrepeatable Read	Phantom Read
READ UNCOMMITTED	possible	possible	possible
READ COMMITTED	not possible	possible	possible
REPEATABLE READ	not possible	not possible	possible
SERIALIZABLE	not possible	not possible	not possible

Tuning of memory

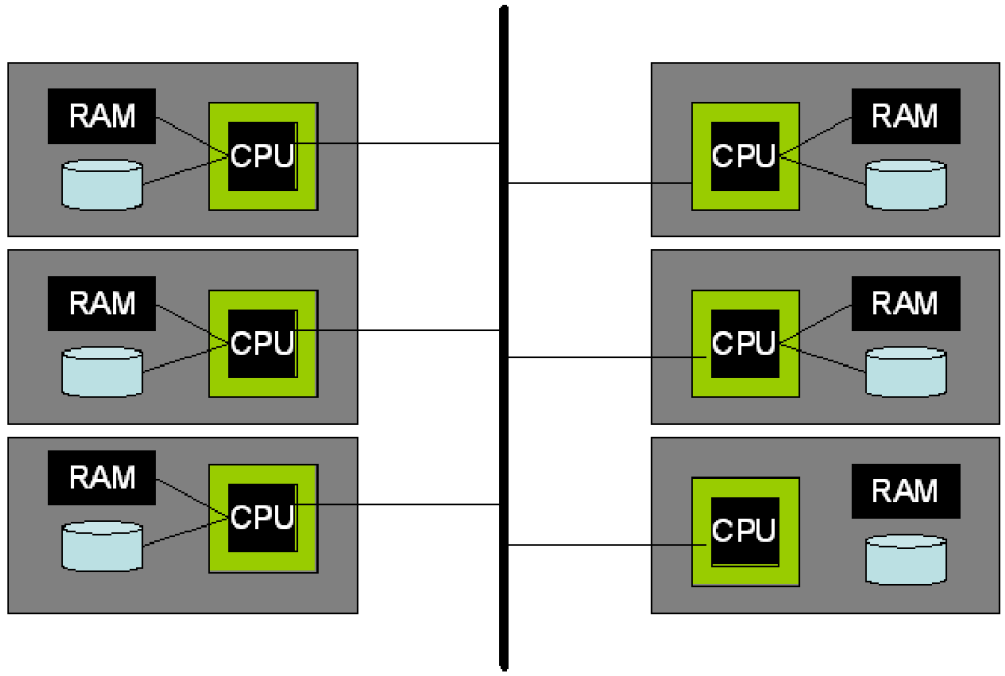
- How to optimize memory allocation?



Oracle Memory Model (Dageville & Zait, VLDB 2002)

Data partitioning

- Increase data availability
- Decrease administrative cost
- Improve query performance



Shared-nothing parallel DBMS (Hellerstein, et al., 2007)

References

Additional Readings:

- J.M. Hellerstein, M. Stonebraker, J. Hamilton, *Architecture of a Database System*, Foundations and Trends in Databases, 1(2), 2007, 141-259.