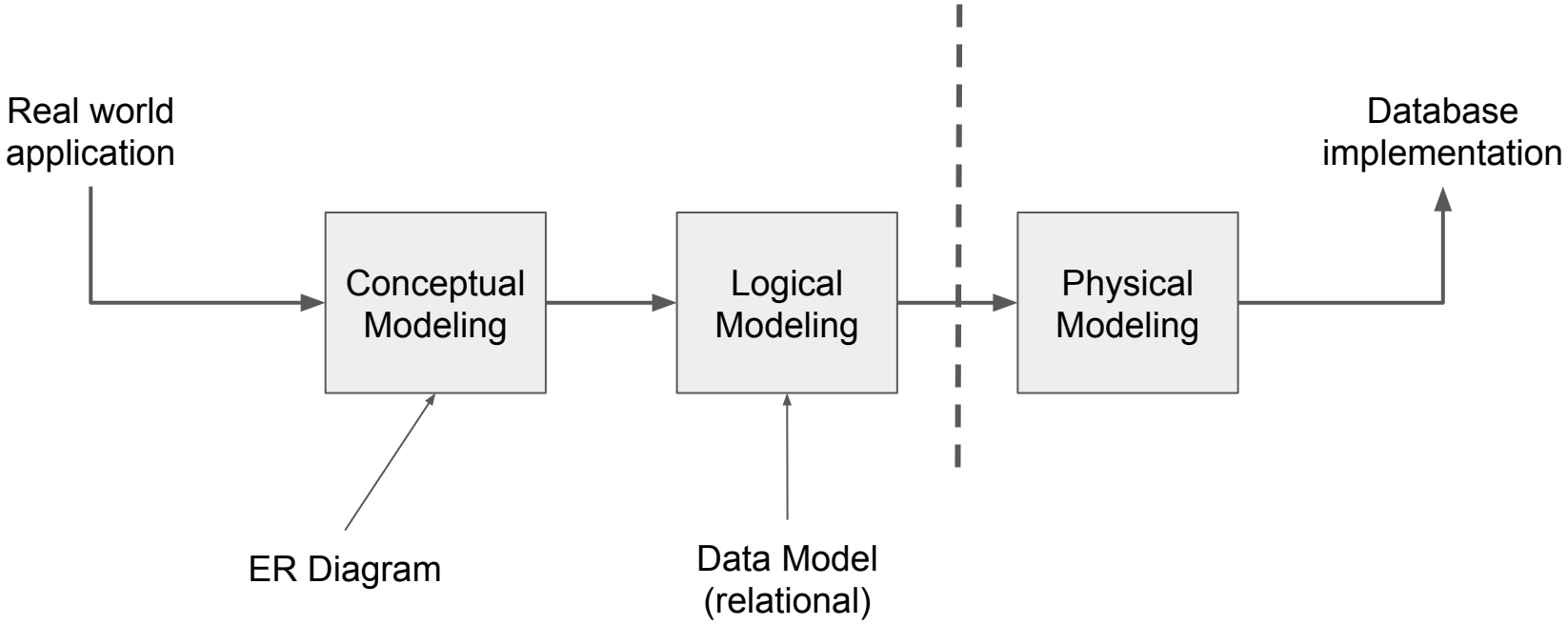
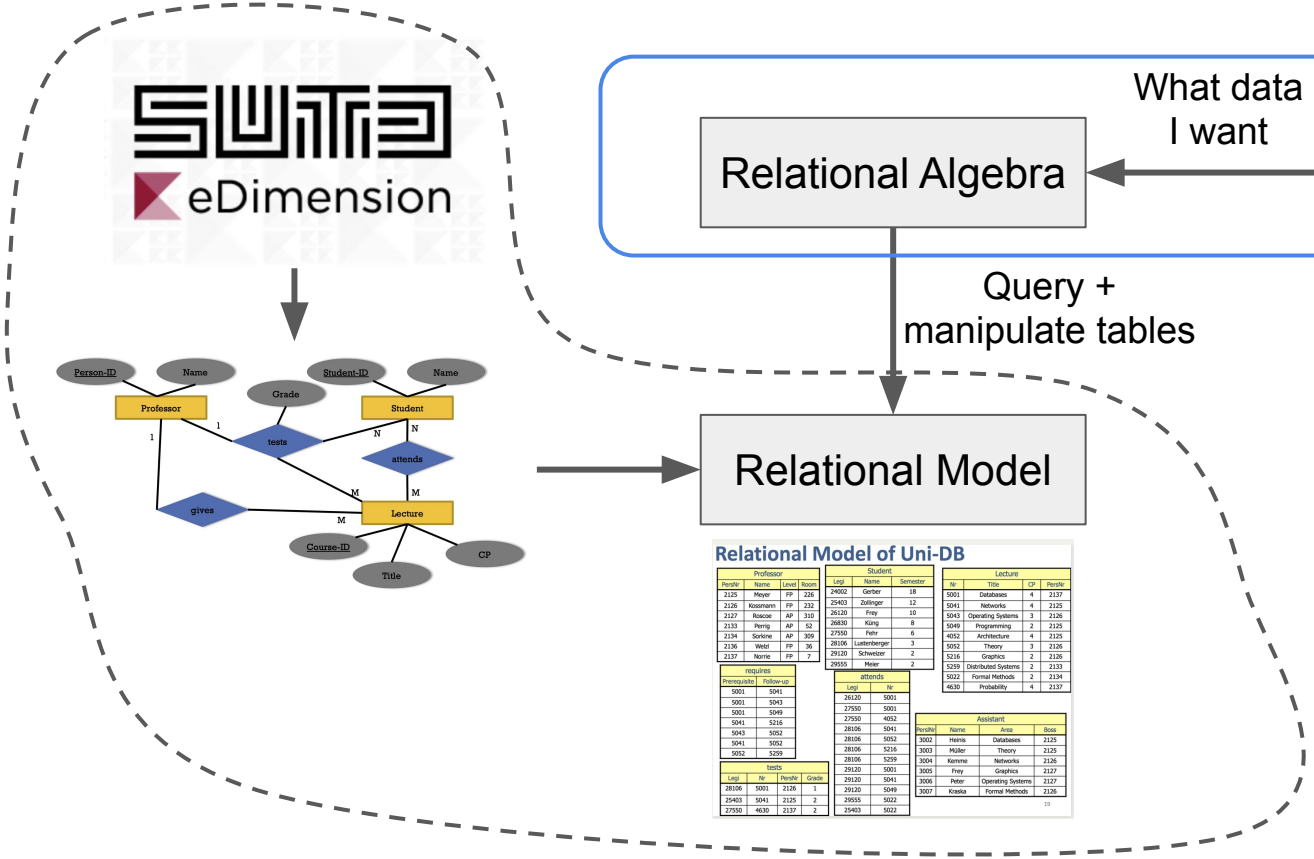
Databases and Big Data

SQL

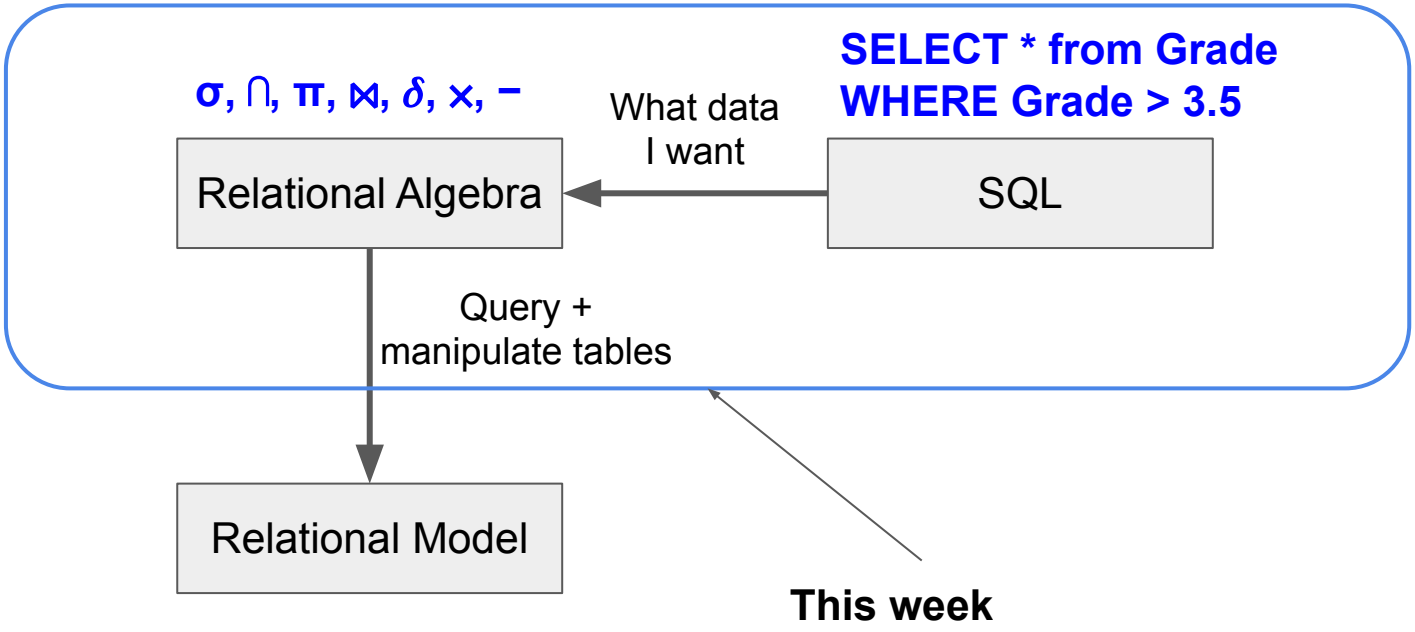
Recap


Real world application
Conceptual Modeling
Logical Modeling
Physical Modeling
Database implementation
ER Diagram
Data Model (relational)

Data Model


eDimension
What data
I want
Relational Algebra
SQL
Query +
manipulate tables
Relational Model
Person-ID
Name
Professor
Grade
tests
Student-ID
Name
Student
attends
gives
Lecture
Course-ID
Title
CP
Relational Model of Uni-DB

Data Model


SELECT * from Grade
WHERE Grade > 3.5
σ, ∩, π, ⋈, δ, ×, −
What data
I want
Relational Algebra
SQL
Query +
manipulate tables
Relational Model
This week

Example

Payroll

UserID	Name	Job	Salary
123	Alice	Asst. Prof	100
456	Bob	TA	80
789	Carol	Asst. Prof	120
101	David	Prof	150

Give me salaries of all Asst. Prof

Here

UserID	Salary
123	100
789	120

Example

SQL query

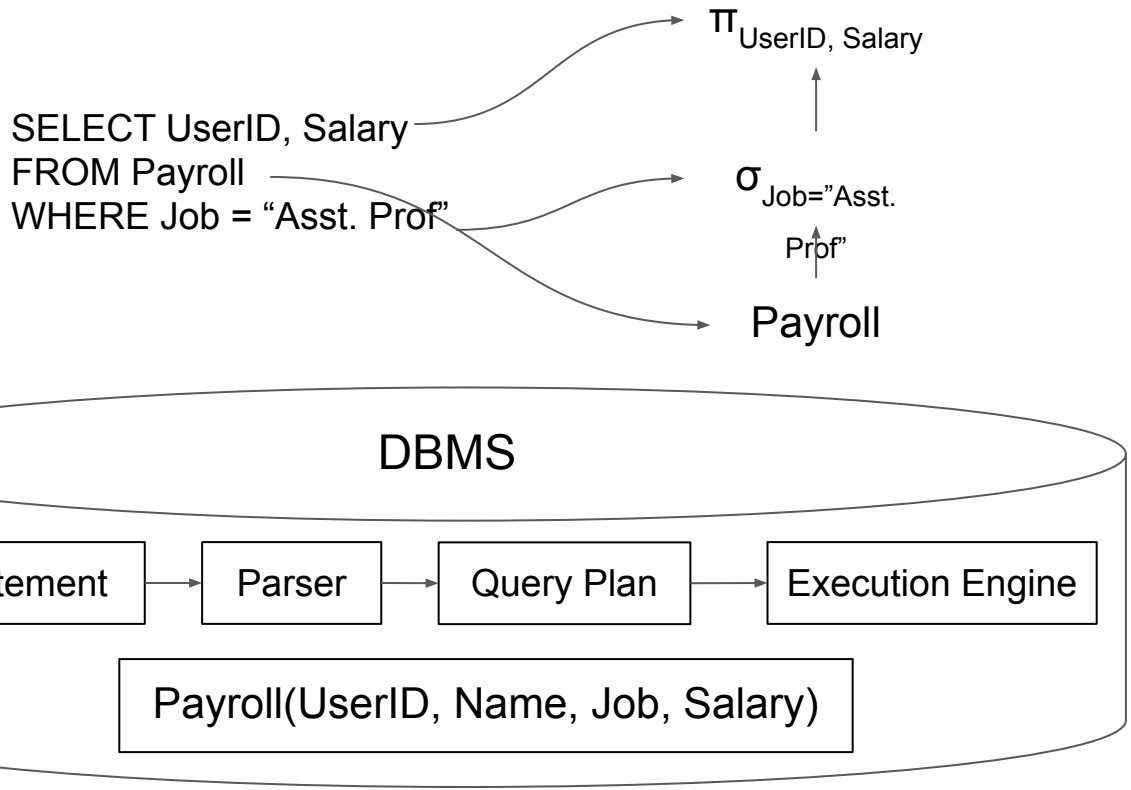
DBMS

Payroll(UserID, Name, Job, Salary)

SELECT UserID, Salary
FROM Payroll
WHERE Job = “Asst. Prof”

Example

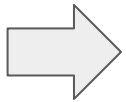
Query Plan
Acts as an algorithm for the DBMS to carry out the operation.

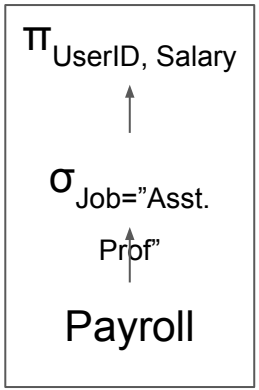


Example

Goes to each row in Payroll table, if the column Job satisfies some condition, then you output the values you want. (This algorithm is a simplified version, real execution engines are much faster)

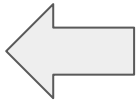
UserID	Name	Job	Salary
123	Alice	Asst. Prof	100
456	Bob	TA	80
789	Carol	Asst. Prof	120
101	David	Prof	150





Execution Engine

```
foreach row in Payroll:
  if (row.Job == "Asst. Prof")
    output (row.UseID, row.Salary)
```



UserID	Salary
123	100
789	120

Relational Algebra

- Algebra:
  - Study of symbols
    - Their meanings
    - Their relationships
- Relational Algebra
  - Study of symbols that manipulates relations
  - Symbols = **operators**

Algebra (from Arabic *"al-jabr"*, literally meaning "reunion of broken parts"[1]) is one of the broad parts of mathematics, together with number theory, geometry and analysis. In its most general form, algebra is the study of mathematical symbols and the rules for manipulating these symbols;[2] it is a unifying thread of almost all of mathematics.[3] It includes everything from elementary equation solving to the study of abstractions such as groups, rings, and

$$x^2 - 2x - 4 = 0$$

$$\sigma_{A>3}(\pi_{X,Y}(R) \bowtie_X \pi_{X,Z}(T)) = S$$

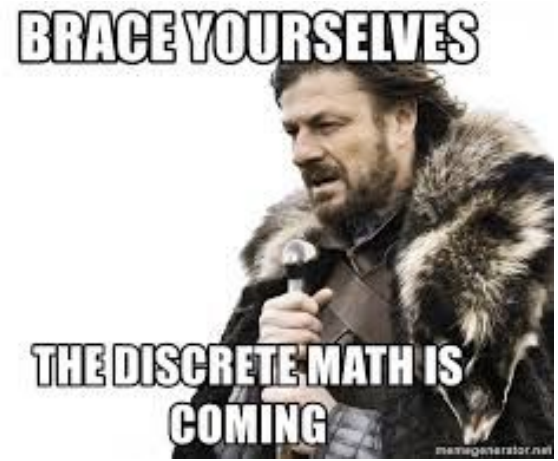
Relational Algebra

- Fundamental operators to:
  - Retrieve
  - Manipulate relations
- Each operator:
  - Take one or more relations as input
  - Output a new relation

σ	Selection
π	Projection
−	Difference
∪	Union
∩	Intersection
⋈	Join
×	Product

Relational Algebra

- We focus on 7 operators
  - They can be chained (or composed) to create more complex operators
- Based on set algebra
  - Because relation is a set



σ	Selection
π	Projection
−	Difference
∪	Union
∩	Intersection
⋈	Join
×	Product

All tuples are unique, proper relation :)

Selection

- Syntax: $\sigma_{predicate}(R)$
- Choose a subset of tuples from R that satisfy the predicate
  - Like a filter
  - Predicate can be complex, with conjunction (AND) and disjunction (OR)

R(A,B)

A	B
a1	101
a2	102
a2	103
a3	104

```
select * from R
where A='a2' and B > 102;
```

$\sigma_{A==”a2”}(R)$

A	B
a2	102
a2	103

$\sigma_{A==”a2” \text{ AND } B>102}(R)$

A	B
a2	103

Projection

- Syntax: $\pi_{A1,A2,..}(R)$
- Generate a relation with tuples containing only the specified attributes
  - Can rearrange attribute order
  - Can transform values

R(A,B)

A	B
a1	101
a2	102
a2	103
a3	104

```
select B-100, A
from R where A='a2';
```

$\pi_{B-100,\ A}(\sigma_{A==\text{"a2"}}(R))$

B-100	A
2	a2
3	a2

Union

Result will be a set as well

- Syntax: **(R ∪ S)**
- Generate a relation with tuples appearing in any of the two relations
  - Exactly like set union

```
(select * from R)
UNION
(select * from S);
```

R(A,B)

A	B
a1	101
a2	102
a3	103

S(A,B)

A	B
a3	103
a4	104
a5	105

R ∪ S

A	B
a1	101
a2	102
a3	103
a4	104
a5	105

Intersection

- Syntax: **(R ∩ S)**
- Generate a relation with tuples appearing in both relations
  - Exactly like set intersection

```
(select * from R)
INTERSECT
(select * from S);
```

R(A,B)

A	B
a1	101
a2	102
a3	103

S(A,B)

A	B
a3	103
a4	104
a5	105

R ∩ S

A	B
a3	103

Difference

- Useful for selecting 2 criteria
select cat='A' - select cat='B' EQUAL select cat='A' AND cat='B'
- Useful for selecting 'only' etc
get all runners who only run 100m category: R - σ(100m)(R)

- Syntax: **(R − S)**
- Generate a relation with tuples appearing in R but not in S
  - Exactly like set difference

```
(select * from R)
EXCEPT
(select * from S);
```

R(A,B)

A	B
a1	101
a2	102
a3	103

S(A,B)

A	B
a3	103
a4	104
a5	105

R − S

A	B
a1	101
a2	102

Product

- Syntax: **(R × S)**
- Generate a relation with all possible combination of tuples from R and S
  - Exactly like set Cartesian product
  - R, S can have different schema

```
select * from R cross join S;
```

R(A,B)

A	B
a1	101
a2	102

S(C,D)

C	D
a3	103
a4	104

A × B

R.A	R.B	S.C	S.D
a1	101	a3	103
a1	101	a4	104
a2	102	a3	103
a2	102	a4	104

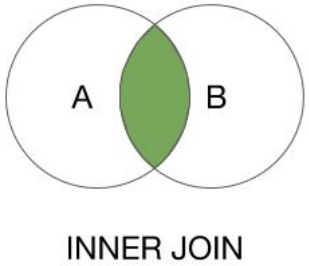
Join

- Bread and butter! Very important
- Already seen cross-join: $\times$
- We focus on three variants:
  - Inner Join (Equi-Join)
  - Natural Join
  - Left/Right/Full Outer Join

After cross product, there will be 6 columns, 9 tuples.
After applying selection, only 2 tuples are returned

Inner Join

- Syntax: $(R \bowtie_{R.A = S.B, R.B=S.D, ..} S)$
- Generate a relation with tuples appearing in $R \times S$ and satisfying condition
  - Product followed by Selection
  - Can join on multiple columns



INNER JOIN

```
select * from R, S
where R.A = S.D;
```

R(A,B,C)

A	B	C
a1	101	0
a2	102	1
a3	103	0

S(D,E,F)

D	E	F
a3	103	‘a’
a1	107	‘b’
a5	105	‘c’

$R \bowtie_{R.A = S.D} S$

R.A	R.B	R.C	S.D	S.E	S.F
a1	101	0	a1	107	‘b’
a3	103	0	a3	103	‘a’

Natural Join

- Syntax: **(R ⋈ S)**
- Like Inner Join, but:
  - Automatically detect **all common attributes** (by names), and use them as join condition
  - Automatically remove duplicate column

```
select * from R natural join S;
```

R(A,B,C)

A	B	C
a1	101	0
a2	102	1
a3	103	0

S(A,E,F)

A	E	F
a3	103	‘a’
a1	107	‘b’
a5	105	‘c’

R ⋈ S

R.A	R.B	R.C	S.E	S.F
a1	101	0	107	‘b’
a3	103	0	103	‘a’

Left Outer Join

- Syntax: $(R \bowtie_{R.A = S.B} S)$
- Same as Inner Join, except:
  - All tuples of R appear in the result

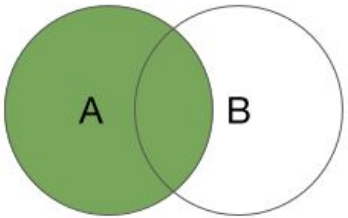
R(A,B,C)

A	B	C
a1	101	0
a2	102	1
a3	103	0

S(D,E,F)

D	E	F
a3	103	'a'
a1	107	'b'
a5	105	'c'

```
select * from R left outer join S
where R.A = S.D;
```



LEFT OUTER JOIN

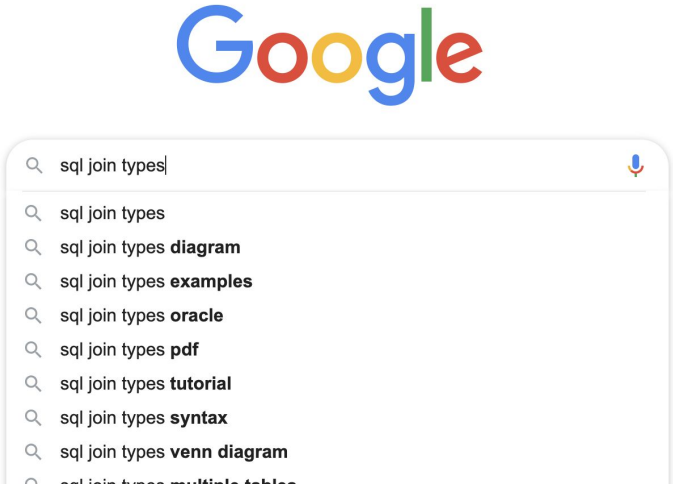
NULL if they don't match with any values in S.
(e.g. a2 does not match with any values in S, it will still appear in the results)

$R \bowtie_{R.A = S.D} S$

R.A	R.B	R.C	S.D	S.E	S.F
a1	101	0	a1	107	'b'
a3	103	0	a3	103	'a'
a2	102	1	*NULL*	*NULL*	*NULL*

Other Operators

- Many other operators
- Check them out yourself



ρ	Rename
δ	Duplicate Elimination
γ	Aggregation
τ	Sorting
$\bowtie$	Right Outer Join
$\bowtie$	Full Outer Join

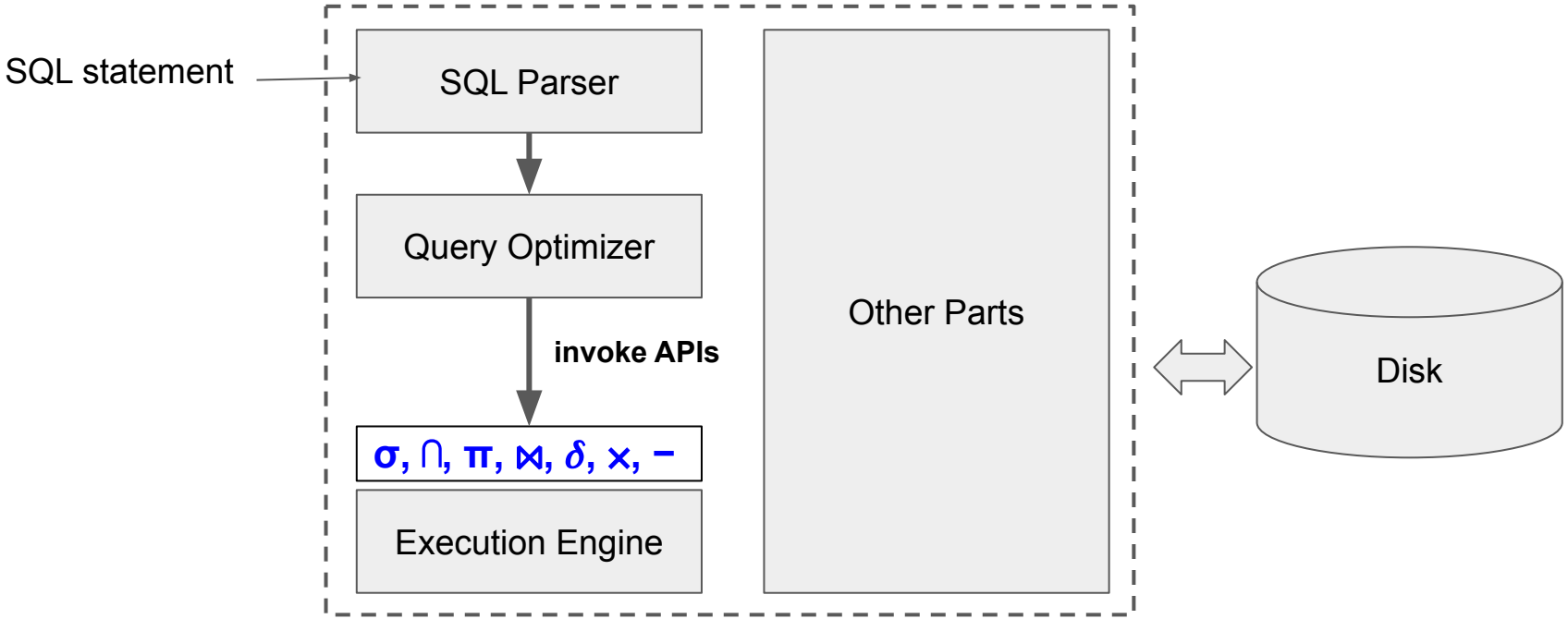
Relational Algebra

- Observation:

$$\sigma_{B=102}(R \bowtie S) = R \bowtie (\sigma_{B=102}(S))$$

- Given input relations, there are > 1 ways to get the desired output
- Good design = only specify the output
  - Let the machine select the best way

Glimpse Into Database Internal



In Class Exercise

R x S *---- 5 columns, 25 tuples*

R natural join S -- 4 columns, 5 tuples

1x03, 2y21, 2y33, 2y20, 3x03

R inner join S (where A=D) -- 5 columns, 3 tuples

1xy21, 3xx03, 3xy33

difference of B in R and B of (C<3) in S

1 column, 2 tuples (z, a)

R(A,B)

A	B
1	x
2	y
2	z
3	x
9	a

S(B,C,D)

B	C	D
x	0	3
y	2	1
y	3	3
w	3	0
y	2	0

Expression	Size of results
$R \times S$	25
$R \bowtie S$	5
$R \bowtie_{A=D} S$	3
$\pi_B(R) - \pi_B(\sigma_{C<3}(S))$	2

In Class Exercise

Reader (ReaderID, FirstName, LastName)
Book (ISBN, Title, Author, PublicationDate, PublisherName)
Publisher (PublisherName, PublisherCity)
Loan (ReaderID, ISBN, Copy, ReturnDate)

Who are the readers who borrow more than 10 copies of a book at a time.

$\Pi_{\text{FirstName, LastName}}(\text{Reader} \bowtie (\sigma_{\text{Copy}>10}(\text{Loan}))$

Which books (Author, Title) are from publishers in New York or London?

Which books did “Anh Dinh” borrowed?

In Class Exercise

Reader (ReaderID, FirstName, LastName)
Book (ISBN, Title, Author, PublicationDate, PublisherName)
Publisher (PublisherName, PublisherCity)
Loan (ReaderID, ISBN, Copy, ReturnDate)

Who are the readers who borrow more than 10 copies of a book at a time.

Which books (Author, Title) are from publishers in New York **or** London?
$\Pi_{\text{Title,Author}}(\sigma_{\text{PublisherCity=='NewYork' OR PublisherCity=='London'}}(\text{Publisher}) \bowtie \text{Book})$

Which books did “Anh Dinh” borrowed?

In Class Exercise

Reader (ReaderID, FirstName, LastName)
Book (ISBN, Title, Author, PublicationDate, PublisherName)
Publisher (PublisherName, PublisherCity)
Loan (ReaderID, ISBN, Copy, ReturnDate)

Who are the readers who borrow more than 10 copies of a book at a time.

Which books (Author, Title) are from publishers in New York or London?

Which books did “Anh Dinh” borrowed?

$\Pi_{\text{Title,Author}}((\text{Loan} \bowtie \sigma_{\text{LastName=='Dinh' AND FirstName=='Anh'}}(\text{Reader})) \bowtie \text{Book})$