# Database Systems
# CSE 414

## Lectures 8: Relational Algebra

## (Ch. 2.4, & 5.1)

# Announcements

- WQ4 is posted and due on Nov. 3, 11pm

- HW2 will be due next Monday 11pm

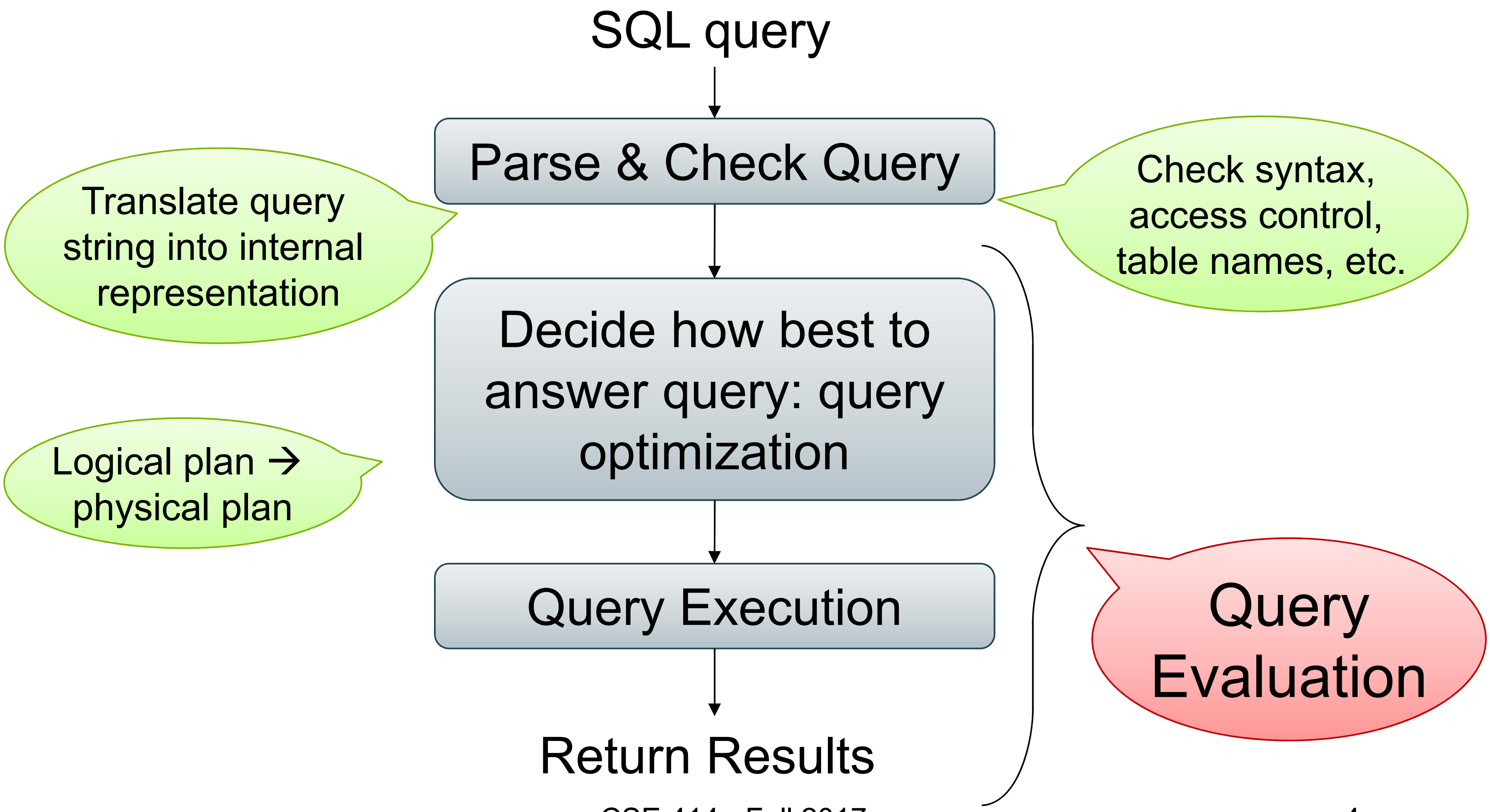
- Log into Azure web site using your outlook.com email address today if you have not done so already
  - Otherwise the TA cannot give you the Azure access code

# Where We Are

- Motivation for using a DBMS for managing data
- SQL:
  - Declaring the schema for our data (CREATE TABLE)
  - Inserting data one row at a time or in bulk (INSERT/.import)
  - Modifying the schema and updating the data (ALTER/UPDATE)
  - Querying the data (SELECT)

- Next step: More knowledge of how DBMSs work
  - Client-server architecture
  - Relational algebra and query execution

# Query Evaluation Steps



SQL query

Parse & Check Query

Decide how best to answer query: query optimization

Query Execution

Return Results

Translate query string into internal representation

Check syntax, access control, table names, etc.

Logical plan → physical plan

Query Evaluation

# The WHAT and the HOW

- SQL = WHAT we want to get from the data
- Relational Algebra = HOW to get the data we want
- Move from WHAT to HOW is query optimization
  - SQL ~> Relational Algebra ~> Physical Plan
  - Relational Algebra = Logical Plan

# Relational Algebra

# Sets vs. Bags

- Sets: {a,b,c}, {a,d,e,f}, { }, . . .
- Bags: {a, a, b, c}, {b, b, b, b, b}, . . .

Relational Algebra has two semantics:

- Set semantics = standard Relational Algebra
- Bag semantics = extended Relational Algebra

DB systems implement bag semantics (Why?)

# Relational Algebra Operators

- Union $\cup$, intersection $\cap$, difference $-$
- Selection $\sigma$
- Projection $\pi$ ($\Pi$)
- Cartesian product $\times$, join $\bowtie$
- Rename $\rho$
- Duplicate elimination $\delta$
- Grouping and aggregation $\gamma$
- Sorting $\tau$

RA

Extended RA

# Union and Difference

$R1 \cup R2$

$R1 - R2$

What do they mean over bags ?

# What about Intersection ?

- Derived operator using minus

$$\boxed{R1 \cap R2 = R1 - (R1 - R2)}$$

- Derived using join (will explain later)

$$\boxed{R1 \cap R2 = R1 \bowtie R2}$$

# Selection

- Returns all tuples that satisfy a condition

$$\boxed{\sigma_c(R)}$$

- Examples
  - $\sigma_{\text{Salary > 40000}}$ (Employee)
  - $\sigma_{\text{name = “Smith”}}$ (Employee)
- The condition c can be =, <, $\leq$, >, $\geq$, <> combined with AND, OR, NOT

Employee

| SSN | Name | Salary |
|---|---|---|
| 1234545 | John | 20000 |
| 5423341 | Smith | 60000 |
| 4352342 | Fred | 50000 |

$\sigma_{\text{Salary > 40000}}$ (Employee)

| SSN | Name | Salary |
|---|---|---|
| 5423341 | Smith | 60000 |
| 4352342 | Fred | 50000 |

# Projection

- Eliminates columns

$$\boxed{\pi_{A1,\ldots,An}(R)}$$

- Example: project social-security number and names:
  - $\Pi_{\text{SSN, Name}}$ (Employee)
  - Answer(SSN, Name)

Different semantics over sets or bags! Why?

Employee

| SSN | Name | Salary |
|---|---|---|
| 1234545 | John | 20000 |
| 5423341 | John | 60000 |
| 4352342 | John | 20000 |

$\pi_{\text{Name, Salary}}$ (Employee)

| Name | Salary |
|---|---|
| John | 20000 |
| John | 60000 |
| John | 20000 |

Bag semantics

| Name | Salary |
|---|---|
| John | 20000 |
| John | 60000 |

Set semantics

Which is more efficient?

# Composing RA Operators

Patient

| no | name | zip | disease |
|---|---|---|---|
| 1 | p1 | 98125 | flu |
| 2 | p2 | 98125 | heart |
| 3 | p3 | 98120 | lung |
| 4 | p4 | 98120 | heart |

$\pi_{\text{zip, disease}}(\text{Patient})$

| zip | disease |
|---|---|
| 98125 | flu |
| 98125 | heart |
| 98120 | lung |
| 98120 | heart |

$\sigma_{\text{disease=‘heart’}}(\text{Patient})$

| no | name | zip | disease |
|---|---|---|---|
| 2 | p2 | 98125 | heart |
| 4 | p4 | 98120 | heart |

$\pi_{\text{zip, disease}}\ (\sigma_{\text{disease=‘heart’}}(\text{Patient}))$

| zip | disease |
|---|---|
| 98125 | heart |
| 98120 | heart |

# Cartesian Product

- Each tuple in R1 with each tuple in R2

$$\boxed{R1 \times R2}$$

- Rare in practice; mainly used to express joins

# Cross-Product Example

**Employee**

| **Name** | **SSN** |
|---|---|
| John | 999999999 |
| Tony | 777777777 |

**Dependent**

| **EmpSSN** | **DepName** |
|---|---|
| 999999999 | Emily |
| 777777777 | Joe |

**Employee × Dependent**

| **Name** | **SSN** | **EmpSSN** | **DepName** |
|---|---|---|---|
| John | 999999999 | 999999999 | Emily |
| John | 999999999 | 777777777 | Joe |
| Tony | 777777777 | 999999999 | Emily |
| Tony | 777777777 | 777777777 | Joe |

# Renaming

- Changes the schema, not the instance

$$\boxed{\rho_{B1,\ldots,Bn}(R)}$$

- Example:
  - $\rho_{N,\,S}(\text{Employee}) \rightarrow \text{Answer}(N, S)$

Not really used by systems, but needed on paper

# Natural Join

$$\boxed{R1 \bowtie R2}$$

- Meaning: $R1 \bowtie R2 = \pi_A(\sigma_\theta(R1 \times R2))$

- Where:
  - Selection $\sigma$ checks equality of all common attributes (attributes with same names)
  - Projection $\pi$ eliminates duplicate common attributes

# Natural Join Example

**R**

| A | B |
|---|---|
| X | Y |
| X | Z |
| Y | Z |
| Z | V |

**S**

| B | C |
|---|---|
| Z | U |
| V | W |
| Z | V |

**R ⋈ S =**

$\pi_{A,B,C}(\sigma_{R.B=S.B}(R \times S))$

| A | B | C |
|---|---|---|
| X | Z | U |
| X | Z | V |
| Y | Z | U |
| Y | Z | V |
| Z | V | W |

# Natural Join Example 2

AnonPatient P

| age | zip | disease |
|---|---|---|
| 54 | 98125 | heart |
| 20 | 98120 | flu |

Voters V

| name | age | zip |
|---|---|---|
| p1 | 54 | 98125 |
| p2 | 20 | 98120 |

$P \bowtie V$

| age | zip | disease | name |
|---|---|---|---|
| 54 | 98125 | heart | p1 |
| 20 | 98120 | flu | p2 |

# Natural Join

- Given schemas R(A, B, C, D), S(A, C, E), what is the schema of R $\bowtie$ S ?
  - (A, B, C, D, E) through join on (A, C)
- Given R(A, B, C), S(D, E), what is R $\bowtie$ S ?
  - (A, B, C, D, E) through cross product
- Given R(A, B), S(A, B), what is R $\bowtie$ S ?
  - (A, B) through intersection

AnonPatient (age, zip, disease)
Voters (name, age, zip)

# Theta Join

- A join that involves a predicate

$$\boxed{R1 \bowtie_{\theta} R2 \quad = \quad \sigma_{\theta} (R1 \times R2)}$$

- Here $\theta$ can be any condition
- For our voters/patients example:

$$P \bowtie_{\text{P.zip = V.zip and P.age >= V.age -1 and P.age <= V.age +1}} V$$

# Equijoin

- A theta join where $\theta$ is an equality predicate
- By far the most used variant of join in practice

# Equijoin Example

AnonPatient P

| age | zip | disease |
|---|---|---|
| 54 | 98125 | heart |
| 20 | 98120 | flu |

Voters V

| name | age | zip |
|---|---|---|
| p1 | 54 | 98125 |
| p2 | 20 | 98120 |

$P \bowtie_{P.age=V.age} V$

| P.age | P.zip | P.disease | P.name | V.zip | V.age |
|---|---|---|---|---|---|
| 54 | 98125 | heart | p1 | 98125 | 54 |
| 20 | 98120 | flu | p2 | 98120 | 20 |

# Join Summary

- **Theta-join**: $R \bowtie_\theta S = \sigma_\theta(R \times S)$
  - Join of R and S with a join condition $\theta$
  - Cross-product followed by selection $\theta$
- **Equijoin**: $R \bowtie_\theta S = \sigma_\theta(R \times S)$
  - Join condition $\theta$ consists only of equalities
- **Natural join**: $R \bowtie S = \pi_A (\sigma_\theta(R \times S))$
  - Equijoin
  - Equality on **all** fields with same name in R and in S
  - Projection $\pi_A$ drops all redundant attributes

# So Which Join Is It ?

When we write R $\bowtie$ S, we usually mean an equijoin, but we often omit the equality predicate when it is clear from the context

# More Joins

- **Outer join**
  - Include tuples with no matches in the output
  - Use NULL values for missing attributes
  - Does not eliminate duplicate columns

- Variants
  - Left outer join
  - Right outer join
  - Full outer join

# Outer Join Example

AnonPatient P

| age | zip | disease |
|---|---|---|
| 54 | 98125 | heart |
| 20 | 98120 | flu |
| 33 | 98120 | lung |

AnnonJob J

| job | age | zip |
|---|---|---|
| lawyer | 54 | 98125 |
| cashier | 20 | 98120 |

P ⟕ J

| P.age | P.zip | disease | job | J.age | J.zip |
|---|---|---|---|---|---|
| 54 | 98125 | heart | lawyer | 54 | 98125 |
| 20 | 98120 | flu | cashier | 20 | 98120 |
| 33 | 98120 | lung | null | null | null |

# More Examples

```
Supplier(sno, sname, scity, sstate)
Part(pno, pname, psize, pcolor)
Supply(sno, pno, qty, price)
```

Name of supplier of parts with size greater than 10

$\pi_{\text{sname}}(\text{Supplier} \bowtie \text{Supply} \bowtie (\sigma_{\text{psize}>10}\ (\text{Part}))$

Name of supplier of red parts or parts with size greater than 10

$\pi_{\text{sname}}(\text{Supplier} \bowtie \text{Supply} \bowtie (\sigma_{\text{psize}>10}\ (\text{Part}) \cup \sigma_{\text{pcolor='red'}}\ (\text{Part})\ )\ )$