The Relational Model

Suan Lee

Database Management System (DBMS)

- Used by all major commercial database systems
- Very simple model
- Query with high-level languages: simple yet expressive
- Efficient implementations

- **Database** = set of named relations (or tables)

- **Database** = set of named relations (or tables)

Student

College

- **Database** = set of named relations (or tables)
- Each relation has a set of named attributes (or columns)

Student

ID	Name	GPA	Photo

College

Name	State	ENR

- **Database** = set of named relations (or tables)
- Each relation has a set of named attributes (or columns)
- Each tuple (or row) has a value for each attribute

Student

ID	Name	GPA	Photo
1	Adam	3.9	
2	Bill	3.4	*NULL*
3	Cathy	*NULL*	
⋮	⋮	⋮	⋮

College

Name	State	ENR
Stanford	CA	15,000
Berkeley	CA	36,000
MIT	MA	10,000
⋮	⋮	⋮

- **Database** = set of named relations (or tables)
- Each relation has a set of named attributes (or columns)
- Each tuple (or row) has a value for each attribute
- Each attribute has a type (or domain)

Student

ID	Name	GPA	Photo
1	Adam	3.9	
2	Bill	3.4	*NULL*
3	Cathy	*NULL*	
⋮	⋮	⋮	⋮

College

Name	State	ENR
Stanford	CA	15,000
Berkeley	CA	36,000
MIT	MA	10,000
⋮	⋮	⋮

- **Schema** = structural description of relations in database
- **Instance** = actual contents at given point in time

Student

ID	Name	GPA	Photo
1	Adam	3.9	
2	Bill	3.4	*NULL*
3	Cathy	*NULL*	
⋮	⋮	⋮	⋮

College

Name	State	ENR
Stanford	CA	15,000
Berkeley	CA	36,000
MIT	MA	10,000
⋮	⋮	⋮

- **NULL** – special value for “unknown” or “undefined”

Student

ID	Name	GPA	Photo
1	Adam	3.9	
2	Bill	3.4	*NULL*
3	Cathy	*NULL*	
⋮	⋮	⋮	⋮

College

Name	State	ENR
Stanford	CA	15,000
Berkeley	CA	36,000
MIT	MA	10,000
⋮	⋮	⋮

- **Key** – attribute whose value is unique in each tuple
  Or set of attributes whose combined values are unique

Student

ID	Name	GPA	Photo
1	Adam	3.9	
2	Bill	3.4	*NULL*
3	Cathy	*NULL*	
⋮	⋮	⋮	⋮

College

Name	State	ENR
Stanford	CA	15,000
Berkeley	CA	36,000
MIT	MA	10,000
⋮	⋮	⋮

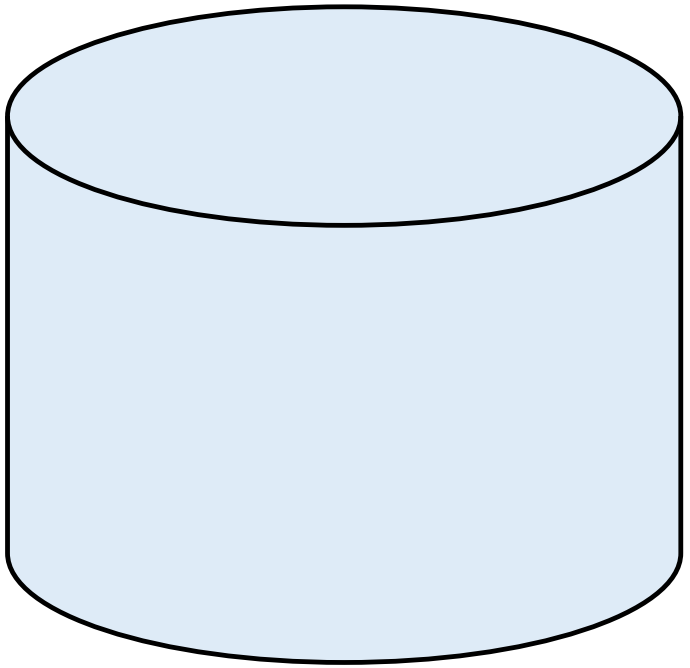
- Creating relations (tables) in SQL

```
CREATE TABLE Student (ID INTEGER, name STRING, GPA FLOAT, photo STRING)

CREATE TABLE College (name STRING, state CHAR(2), enrollment INTEGER)
```

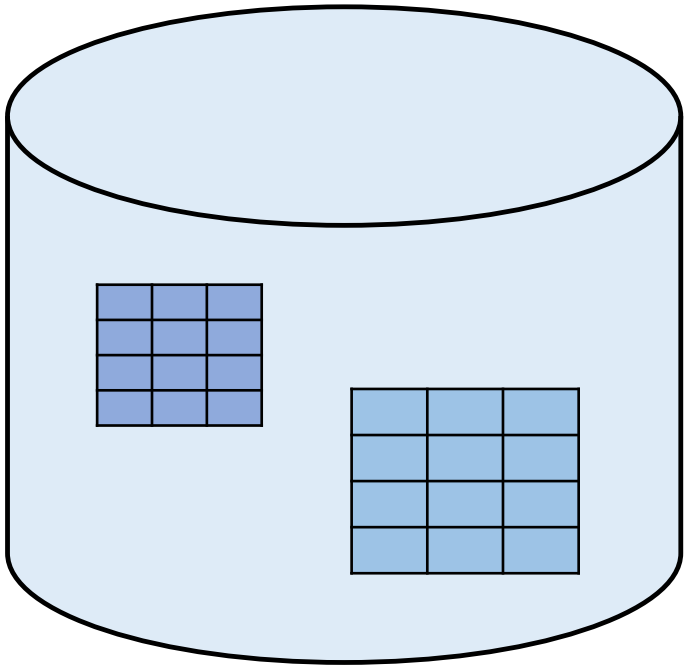
- Used by all major commercial database systems
- Very simple model
- Query with high-level languages: simple yet expressive
- Efficient implementations

Steps in creating and using a (relational) database



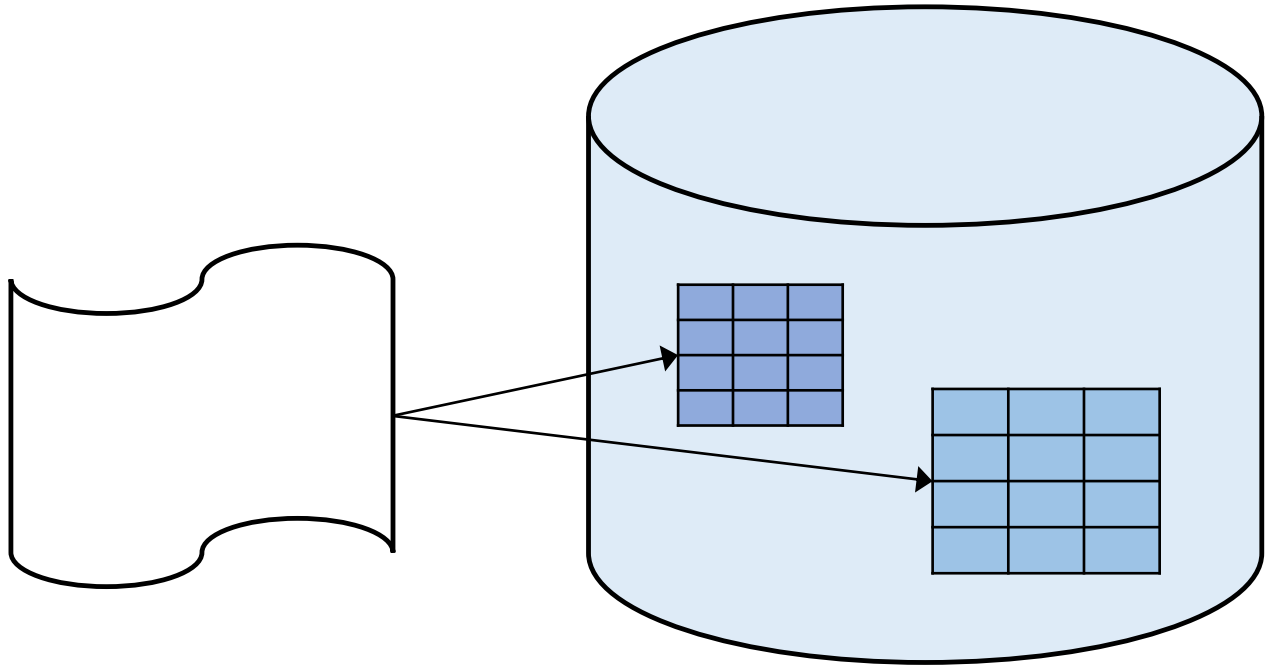
Steps in creating and using a (relational) database

1. Design schema; create using DDL



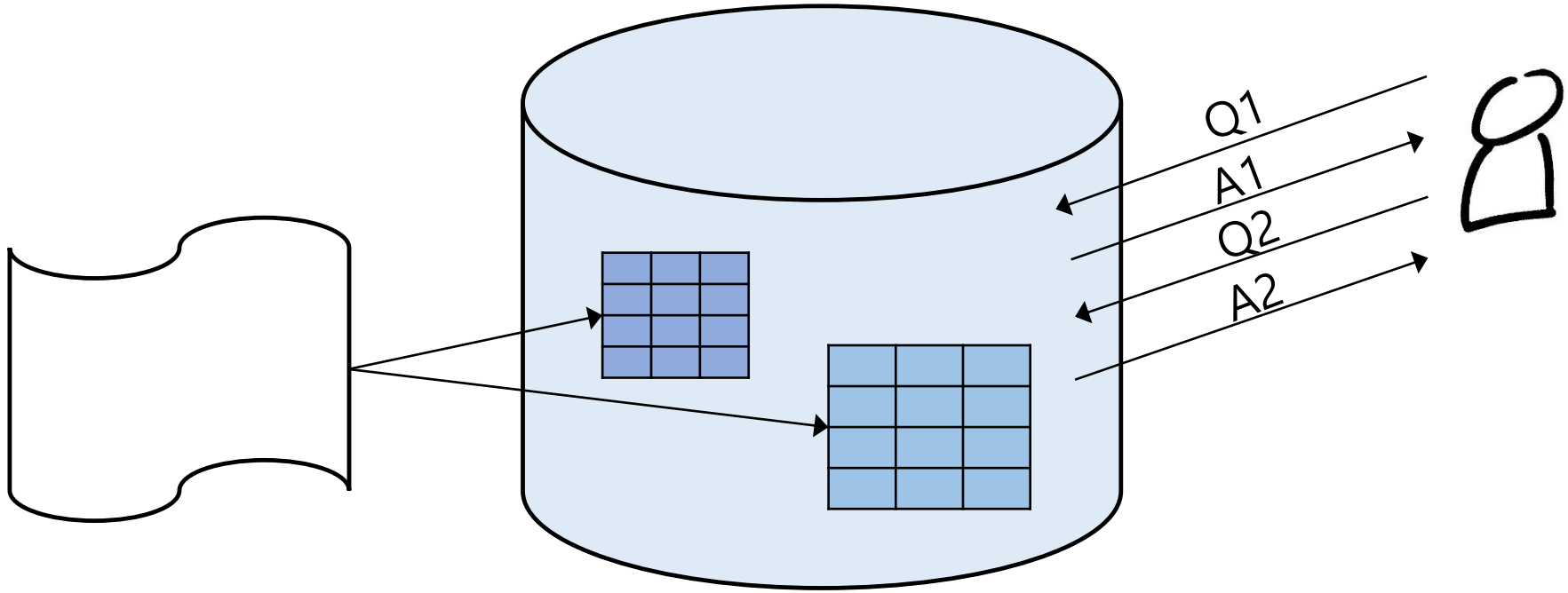
Steps in creating and using a (relational) database

1. Design schema; create using DDL
2. “Bulk load” initial data



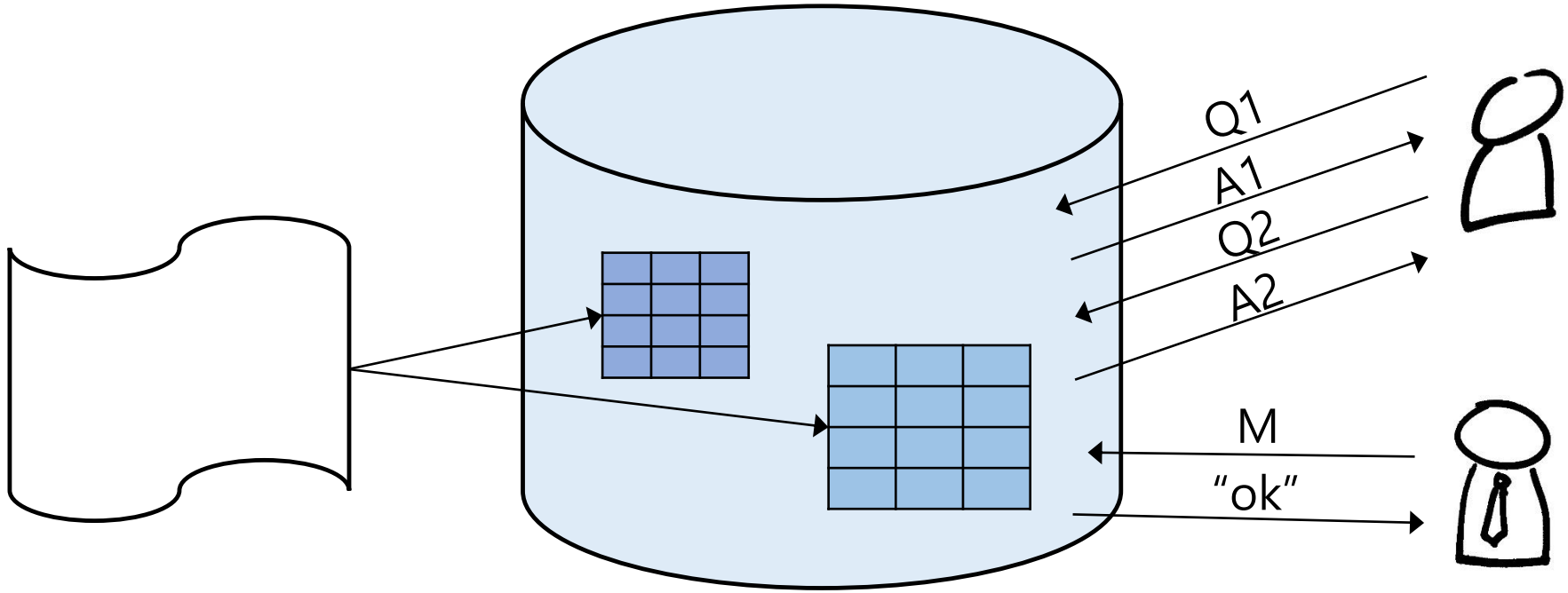
Steps in creating and using a (relational) database

1. Design schema; create using DDL
2. “Bulk load” initial data
3. Repeat: execute queries and modifications



Steps in creating and using a (relational) database

1. Design schema; create using DDL
2. “Bulk load” initial data
3. Repeat: execute queries and modifications



Ad-hoc queries in high-level language

Ad-hoc queries in high-level language

All students with GPA > 3.7 applying to Stanford and MIT only

All engineering departments in CA with < 500 applicants

College with highest average accept rate over last 5 years

Ad-hoc queries in high-level language

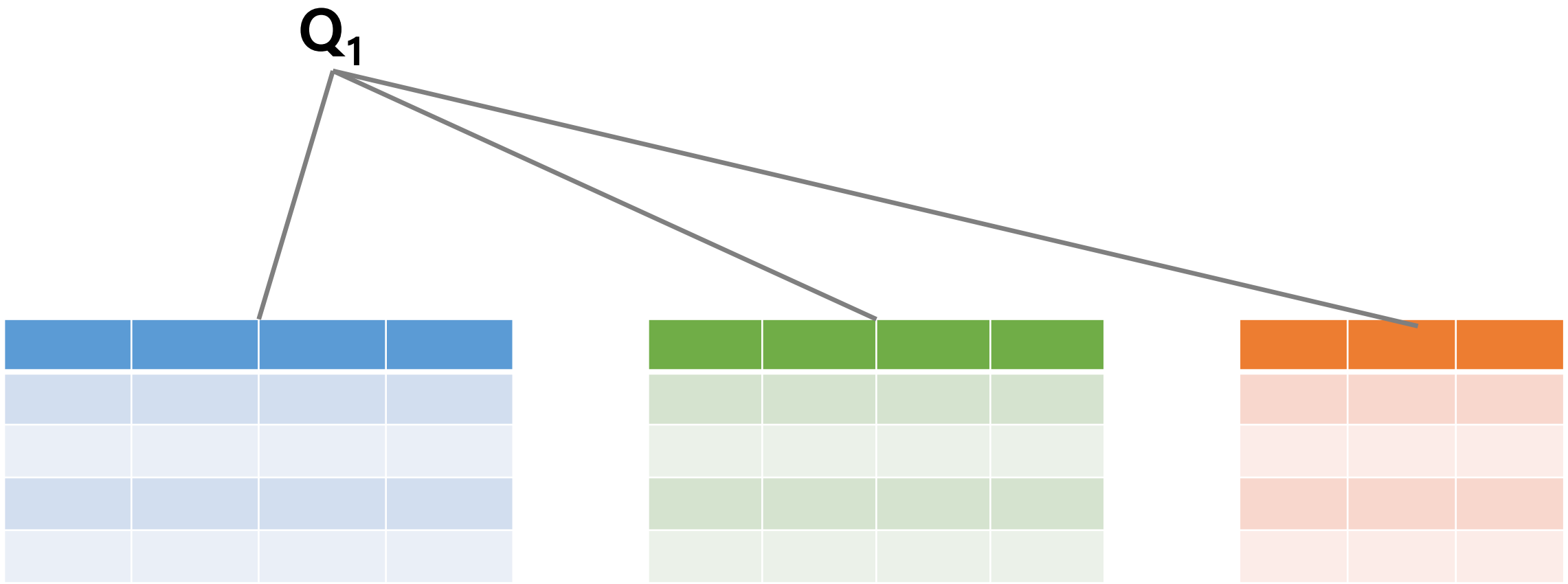
All students with GPA > 3.7 applying to Stanford and MIT only

All engineering departments in CA with < 500 applicants

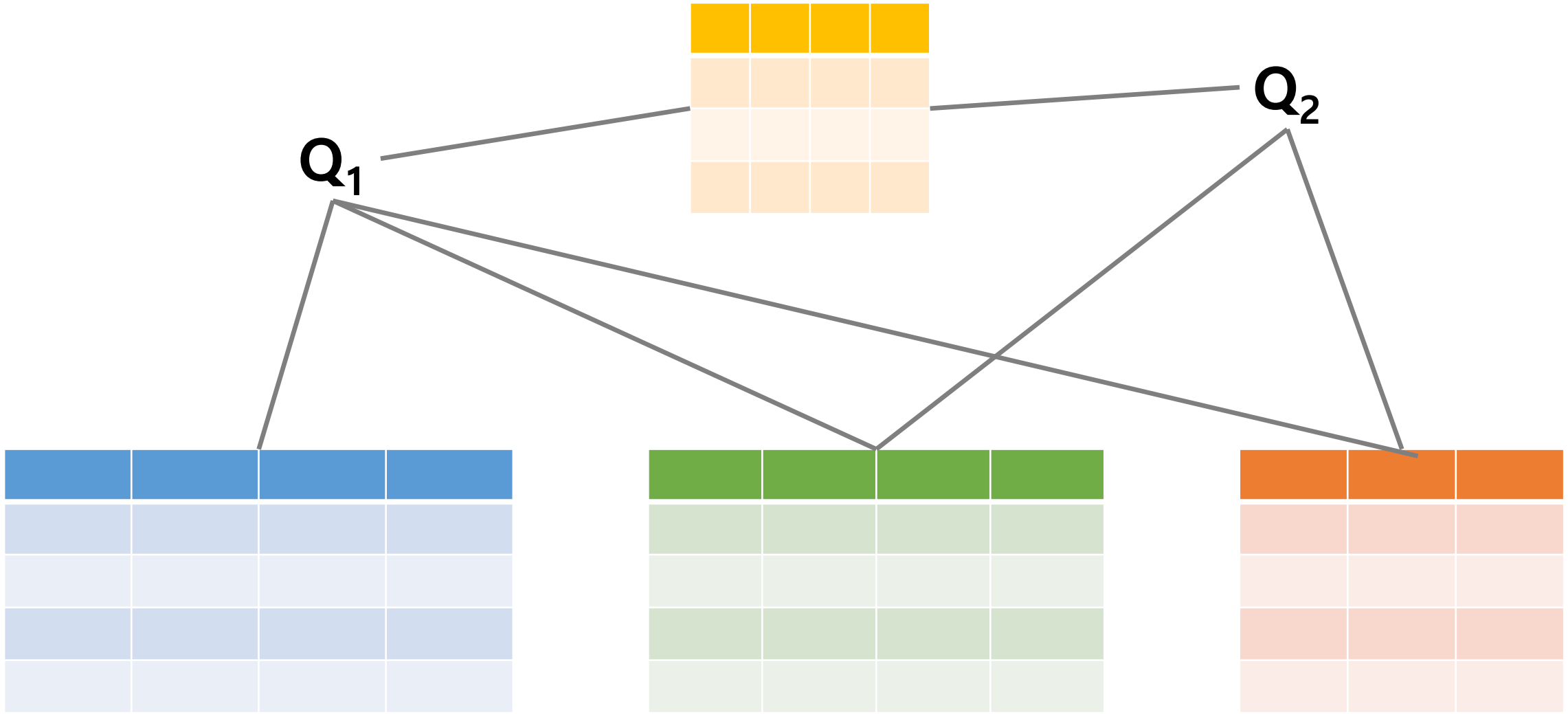
College with highest average accept rate over last 5 years

- Some easy to pose; some a bit harder
- Some easy for DBMS to execute efficiently; some harder
- "Query language" also used to modify data

Queries return relations (“compositional”, “closed”)



Queries return relations ("compositional", "closed")



Query Languages

- Relational Algebra (formal)

- SQL (actual/implemented)

Query Languages

- Relational Algebra (formal)

- SQL (actual/implemented)

IDs of students with GPA > 3.7 applying to Stanford

Query Languages

- Relational Algebra (formal)

$$\Pi_{ID}(\sigma_{GPA=3.7 \wedge College.Name='Stanford'}(Student \bowtie Apply))$$

- SQL (actual/implemented)

IDs of students with GPA > 3.7 applying to Stanford

Query Languages

- Relational Algebra (formal)

$$\Pi_{ID}(\sigma_{GPA=3.7 \wedge College.Name='Stanford'}(Student \bowtie Apply))$$

- SQL (actual/implemented)

```
SELECT Student.ID
FROM   Student, Apply
WHERE  Student.ID=Apply.ID
AND GPA>3.7 AND College.name='Stanford'
```

IDs of students with GPA > 3.7 applying to Stanford