UCLA

CS143
Basic SQL

Professor Junghoo “John” Cho

SQL (Structured Query Language)

- *The* query language for RDBMS
- SQL has many aspects
  - DDL, DML, transactions, …
- In this lecture, we learn DML part of SQL
  - How to query and modify existing database
- SQL and DBMS
  - SQL is a high-level description of what a user wants
  - Given SQL query, DBMS figures out how best to execute it *automatically*
    - Beauty and success of DBMS

Example Database: School Information

Student(sid, name, addr, age, GPA)

sid	name	addr	age	GPA
301	Andy	183 Westwood	19	2.1
303	Elaine	301 Wilshire	17	3.9
401	James	183 Westwood	17	3.5
208	Esther	421 Wilshire	20	3.1

Class(dept, cnum, sec, unit, title, instructor)

dept	cnum	sec	unit	title	instructor
CS	112	01	03	Modeling	Dick Muntz
CS	143	01	04	DB Systems	John Cho
EE	143	01	03	Signal	Dick Muntz
ME	183	02	05	Mechanics	Susan Tracey

Enroll(sid, dept, cnum, sec)

sid	dept	cnum	sec
301	CS	112	01
301	CS	143	01
303	EE	143	01
303	CS	112	01
401	CS	112	01

Q1: Titles and instructors of all CS classes

Student(sid, name, addr, age, GPA)

sid	name	addr	age	GPA
301	Andy	183 Westwood	19	2.1
303	Elaine	301 Wilshire	17	3.9
401	James	183 Westwood	17	3.5
208	Esther	421 Wilshire	20	3.1

Class(dept, cnum, sec, unit, title, instructor)

dept	cnum	sec	unit	title	instructor
CS	112	01	03	Modeling	Dick Muntz
CS	143	01	04	DB Systems	John Cho
EE	143	01	03	Signal	Dick Muntz
ME	183	02	05	Mechanics	Susan Tracey

Enroll(sid, dept, cnum, sec)

sid	dept	cnum	sec
301	CS	112	01
301	CS	143	01
303	EE	143	01
303	CS	112	01
401	CS	112	01

```
SELECT title, instructor
FROM Class
WHERE dept='CS';
```

Basic SQL SELECT statement

DISTINCT

- SELECT $A_1, \ldots, A_n$
  FROM $R_1, \ldots, R_m$
  WHERE C
  $\cong \pi_{A_1,\ldots,A_n}(\sigma_C(R_1 \times \ldots \times R_m))$

clause

SELECT *
FROM Student;

- SELECT *: all attributes
- Note
  - SELECT is "projection" not "selection": can be confusing!
  - ***SQL does not remove duplicates***: main difference between SQL and relational algegra
    - ***Multiset semantics*** for SQL, ***set semantics*** for relational algebra

=bag

Q2: Names and GPAs of all students who take CS class(es)

Student(sid, name, addr, age, GPA)

sid	name	addr	age	GPA
301	Andy	183 Westwood	19	2.1
303	Elaine	301 Wilshire	17	3.9
401	James	183 Westwood	17	3.5
208	Esther	421 Wilshire	20	3.1

Class(dept, cnum, sec, unit, title, instructor)

dept	cnum	sec	unit	title	instructor
CS	112	01	03	Modeling	Dick Muntz
CS	143	01	04	DB Systems	John Cho
EE	143	01	03	Signal	Dick Muntz
ME	183	02	05	Mechanics	Susan Tracey

Enroll(sid, dept, cnum, sec)

sid	dept	cnum	sec
301	CS	112	01
301	CS	143	01
303	EE	143	01
303	CS	112	01
401	CS	112	01

```
SELECT DISTINCT name, GPA grade
FROM Enroll AS E, Student S
WHERE dept='CS' AND E.sid = S.sid;
```

More on Q2

- SELECT name, GPA
  FROM Student S, Enroll E
  WHERE S.sid = E.sid AND dept='CS'
- S, E: tuple variable
  - "renaming operator" in relational algebra
  - S and E are "variables" that bind to every tuple pair from Student and Enroll
- Attributes can be renamed
  - GPA (AS) grade
- DISTINCT: remove duplicates in the result

Q3: All student names and GPAs who live on Wilshire

Student(sid, name, addr, age, GPA)

sid	name	addr	age	GPA
301	Andy	183 Westwood	19	2.1
303	Elaine	301 Wilshire	17	3.9
401	James	183 Westwood	17	3.5
208	Esther	421 Wilshire	20	3.1

Class(dept, cnum, sec, unit, title, instructor)

dept	cnum	sec	unit	title	instructor
CS	112	01	03	Modeling	Dick Muntz
CS	143	01	04	DB Systems	John Cho
EE	143	01	03	Signal	Dick Muntz
ME	183	02	05	Mechanics	Susan Tracey

Enroll(sid, dept, cnum, sec)

sid	dept	cnum	sec
301	CS	112	01
301	CS	143	01
303	EE	143	01
303	CS	112	01
401	CS	112	01

```
SELECT name, GPA
FROM Student
WHERE addr LIKE '%Wilshire%';
```

More on Q3

logical equivalence of query

- %: any length string (0 - ∞)
  _: one character
  '%Wilshire%': any string containing Wilshire
- Q: What does '_ _ _%' mean? '%___'

addr LIKE '___'

addr LIKE '___%'

- Common string functions exist
  - UPPER(), LOWER(), CONCAT(), …

Set Operators

- SQL Set operators: UNION, INTERSECT, EXCEPT
  - Can be applied to relations or to the result of SELECT statements
- Schemas of input relations should be the same
  - In practice, just having the compatible types is fine
- Set operators follow set semantics and remove duplicates
  - Most people do now know “multiset” semantic of set operators
  - No efficiency penalty for duplicate eliminate for set operation
  - To keep duplicates, use UNION ALL, INTERSECT ALL, EXCEPT ALL

Q4: Students' and instructors' names

Student(sid, name, addr, age, GPA)

sid	name	addr	age	GPA
301	Andy	183 Westwood	19	2.1
303	Elaine	301 Wilshire	17	3.9
401	James	183 Westwood	17	3.5
208	Esther	421 Wilshire	20	3.1

Class(dept, cnum, sec, unit, title, instructor)

dept	cnum	sec	unit	title	instructor
CS	112	01	03	Modeling	Dick Muntz
CS	143	01	04	DB Systems	John Cho
EE	143	01	03	Signal	Dick Muntz
ME	183	02	05	Mechanics	Susan Tracey

Enroll(sid, dept, cnum, sec)

sid	dept	cnum	sec
301	CS	112	01
301	CS	143	01
303	EE	143	01
303	CS	112	01
401	CS	112	01

```
(SELECT name
 FROM Student)
UNION
(SELECT instructor name
 FROM Class);
```

Q5: Sids of students who do not take any CS class

Student(sid, name, addr, age, GPA)

sid	name	addr	age	GPA
301	Andy	183 Westwood	19	2.1
303	Elaine	301 Wilshire	17	3.9
401	James	183 Westwood	17	3.5
208	Esther	421 Wilshire	20	3.1

Class(dept, cnum, sec, unit, title, instructor)

dept	cnum	sec	unit	title	instructor
CS	112	01	03	Modeling	Dick Muntz
CS	143	01	04	DB Systems	John Cho
EE	143	01	03	Signal	Dick Muntz
ME	183	02	05	Mechanics	Susan Tracey

Enroll(sid, dept, cnum, sec)

sid	dept	cnum	sec
301	CS	112	01
301	CS	143	01
303	EE	143	01
303	CS	112	01
401	CS	112	01

```
SELECT sid FROM Student)
EXCEPT
(SELECT sid
FROM Enroll
WHERE dept='CS');
```

Set Operators in MySQL

- MySQL supports only UNION, but not INTERSECT or EXCEPT
  - A major pain point since EXCEPT is an essential operator
  - People often use a “subquery” to simulate EXCEPT
    - Use “NOT IN” operator in MySQL that we will learn soon
- MariaDB supports INTERSECT and EXCEPT (starting from v10.3)
  - Our container uses MariaDB

Subqueries

- SELECT statement may appear inside another SELECT statement
  - “Nested” SELECT statements
- Interpretation of subquery
  - The result from inner SELECT statement is treated like a regular relation
  - *Scalar-valued subquery*: If the result is one-attribute one-tuple relation, the result can be used like a ‘constant value’

Q6: Sids who live with student 301

Student(sid, name, addr, age, GPA)

sid	name	addr	age	GPA
301	Andy	183 Westwood	19	2.1
303	Elaine	301 Wilshire	17	3.9
401	James	183 Westwood	17	3.5
208	Esther	421 Wilshire	20	3.1

Class(dept, cnum, sec, unit, title, instructor)

dept	cnum	sec	unit	title	instructor
CS	112	01	03	Modeling	Dick Muntz
CS	143	01	04	DB Systems	John Cho
EE	143	01	03	Signal	Dick Muntz
ME	183	02	05	Mechanics	Susan Tracey

Enroll(sid, dept, cnum, sec)

sid	dept	cnum	sec
301	CS	112	01
301	CS	143	01
303	EE	143	01
303	CS	112	01
401	CS	112	01

```
SELECT sid
FROM Student
WHERE addr = (SELECT addr
              FROM Student
              WHERE sid = 301);
```

$$\pi_{sid}\left(\sigma_{addr = \pi_{addr}(\sigma_{sid=301}(Student))}(Student)\right)$$

Unnesting Subquery

- Q: Can we rewrite Q6 without using subquery?

```
SELECT sid FROM Student WHERE addr =
        (SELECT addr FROM Student WHERE sid=301);
```

```
SELECT   S2.sid
FROM   Student S1, Student  S2
WHERE  S1.srd = 301  AND  S1.addr = S2.addr ;
```

Unnesting Subquery

- A large body of theory and algorithms exist on how to “unnest” a subquery to non-subquery SQL
  - We can rewrite subqueries to non-subqueries as long as there is no negation (NOT)
  - With negation, we need EXCEPT
- Another demonstration of the success of relational model
  - Simple theoretical model makes it possible to create important theorems and algorithms

Q7: Student names who take CS classes

Student(sid, name, addr, age, GPA)

sid	name	addr	age	GPA
301	Andy	183 Westwood	19	2.1
303	Elaine	301 Wilshire	17	3.9
401	James	183 Westwood	17	3.5
208	Esther	421 Wilshire	20	3.1

Class(dept, cnum, sec, unit, title, instructor)

dept	cnum	sec	unit	title	instructor
CS	112	01	03	Modeling	Dick Muntz
CS	143	01	04	DB Systems	John Cho
EE	143	01	03	Signal	Dick Muntz
ME	183	02	05	Mechanics	Susan Tracey

Enroll(sid, dept, cnum, sec)

sid	dept	cnum	sec
301	CS	112	01
301	CS	143	01
303	EE	143	01
303	CS	112	01
401	CS	112	01

{301, 303, 401}

```
SELECT name
FROM Student
WHERE sid IN (SELECT
              FROM Enroll
              WHERE dept='CS');
```

{301, 301, 401, 303}
sid

set membership operator
NOT IN

Subquery: Set Membership Operator

- IN, NOT IN
  - (a IN R) is TRUE if a appears in R

- Q: Can we write Q7 without subqueries?

```
SELECT DISTINCT name
FROM Student S, Enroll E
WHERE S.sid = E.sid AND E.dept = 'CS';
```

- Q: Are the two queries equivalent?

Q8: Student names who take no CS class

Student(sid, name, addr, age, GPA)

sid	name	addr	age	GPA
301	Andy	183 Westwood	19	2.1
303	Elaine	301 Wilshire	17	3.9
401	James	183 Westwood	17	3.5
208	Esther	421 Wilshire	20	3.1

Class(dept, cnum, sec, unit, title, instructor)

dept	cnum	sec	unit	title	instructor
CS	112	01	03	Modeling	Dick Muntz
CS	143	01	04	DB Systems	John Cho
EE	143	01	03	Signal	Dick Muntz
ME	183	02	05	Mechanics	Susan Tracey

Enroll(sid, dept, cnum, sec)

sid	dept	cnum	sec
301	CS	112	01
301	CS	143	01
303	EE	143	01
303	CS	112	01
401	CS	112	01

replace IN with NOT IN in Q7

```
(SELECT name FROM Student)
EXCEPT
(SELECT name FROM Student S, Enroll E
 WHERE S.sid = E.std AND dept = 'CS');
```

Q: Can we rewrite it without subqueries?

Q9: Student IDs who has higher GPA than any student of age 18 or less

Student(sid, name, addr, age, GPA)

sid	name	addr	age	GPA
301	Andy	183 Westwood	19	2.1
303	Elaine	301 Wilshire	17	3.9
401	James	183 Westwood	17	3.5
208	Esther	421 Wilshire	20	3.1

Class(dept, cnum, sec, unit, title, instructor)

dept	cnum	sec	unit	title	instructor
CS	112	01	03	Modeling	Dick Muntz
CS	143	01	04	DB Systems	John Cho
EE	143	01	03	Signal	Dick Muntz
ME	183	02	05	Mechanics	Susan Tracey

Enroll(sid, dept, cnum, sec)

sid	dept	cnum	sec
301	CS	112	01
301	CS	143	01
303	EE	143	01
303	CS	112	01
401	CS	112	01

```
SELECT sid
FROM Student
WHERE GPA >ALL (SELECT GPA
                FROM Student
                WHERE age <= 18) ;
```

set comparison

>= ALL
= ALL

∀ > SOME
∃

Subquery: Set Comparison Operator

- (a > ALL R), (a <= SOME R), …: Compare a against tuples in R
  - “a > ALL R” is TRUE if a is larger than all tuples in R
- Q: Is “= SOME” equivalent to IN?

Q10: Student IDs who has higher GPA than at least one student of age 18 or less

Student(sid, name, addr, age, GPA)

sid	name	addr	age	GPA
301	Andy	183 Westwood	19	2.1
303	Elaine	301 Wilshire	17	3.9
401	James	183 Westwood	17	3.5
208	Esther	421 Wilshire	20	3.1

Class(dept, cnum, sec, unit, title, instructor)

dept	cnum	sec	unit	title	instructor
CS	112	01	03	Modeling	Dick Muntz
CS	143	01	04	DB Systems	John Cho
EE	143	01	03	Signal	Dick Muntz
ME	183	02	05	Mechanics	Susan Tracey

Enroll(sid, dept, cnum, sec)

sid	dept	cnum	sec
301	CS	112	01
301	CS	143	01
303	EE	143	01
303	CS	112	01
401	CS	112	01

replace ALL with SOME in Q9

Q6: Student names who take any class

Student(sid, name, addr, age, GPA)

sid	name	addr	age	GPA
301	Andy	183 Westwood	19	2.1
303	Elaine	301 Wilshire	17	3.9
401	James	183 Westwood	17	3.5
208	Esther	421 Wilshire	20	3.1

Class(dept, cnum, sec, unit, title, instructor)

dept	cnum	sec	unit	title	instructor
CS	112	01	03	Modeling	Dick Muntz
CS	143	01	04	DB Systems	John Cho
EE	143	01	03	Signal	Dick Muntz
ME	183	02	05	Mechanics	Susan Tracey

Enroll(sid, dept, cnum, sec)

sid	dept	cnum	sec
301	CS	112	01
301	CS	143	01
303	EE	143	01
303	CS	112	01
401	CS	112	01

correlated subquery

Correlated Subquery

{Andy, James, Elaine}

- SELECT name
  FROM Student S
  WHERE EXISTS(SELECT * FROM Enroll E WHERE E.sid = S.sid)
- Conceptually, this is how correlated subquery is executed:
  - Outer query looks at one tuple at a time and binds to the tuple to S
  - For each S, we execute the inner query and check the condition
- This is just conceptual description
  - Many DBMS executes it more efficiently than the above description while producing the same result

Subquery in FROM

- SELECT name
  FROM (SELECT name, age FROM Student) S
  WHERE age > 17
- A subquery inside FROM ***must*** be renamed
- Q: Does subquery give SQL more expressive power than relational algebra?

Common Table Expression (SQL99)

- WITH (alias) AS (subquery)
  SELECT … FROM alias …
  - Very convenient for using the same subquery multiple times
- Example:

  ```
  WITH S AS (SELECT name, age FROM Student)
  SELECT name FROM S WHERE age > 17
  ```

What We Learned: Basic SELECT Query

- SELECT … FROM … WHERE
  - Multiset semantic: Duplicates are preserved unless DISTINCT
- Set operator
- Subqueries
  - Scalar-valued subquery
  - Set membership
  - Set comparison
  - Correlated subquery
- Common table expression