In the Lecture Series Introduction to Database Systems

SQL

Presented by Stéphane Bressan

Using SQLite for the Lecture

```
.open cs2102.db
.mode column
.header on
.read small_data.sql
```

Querying One Table

Find all the information about students.

We can use either * or list the columns that we want to retrieve in the SELECT clause.

```
SELECT *
FROM student;

SELECT name, email, year, faculty, department, graduate
FROM student;
```

name	email	year	faculty	department	graduate
GOH HUI YING	gohhuiying1989@gmail.com	1/1/2008	Faculty of Science	Biology	1/1/2012
TAY WEI GUO	tayweiguo1989@msn.com	8/1/2010	Faculty of Engineering	CE	1/1/2013
PENG JIAYUAN	pengjiayuan2011@hotmail.com	1/1/2008	Faculty of Science	Biology	
HUANG ZHANPENG	huangzhanpeng1992@msn.com	8/1/2010	Faculty of Arts and Social Science	Geography	
ZHENG ZHEMIN	zhengzhemin1991@yahoo.com	8/1/2008	Faculty of Arts and Social Science	History	
…					

Selecting Columns

Find the names and emails of students.

WE can selectively choose the columns we want to retrieve in the SELECT clause.

```
SELECT name, email
FROM student;
```

name	email
GOH HUI YING	gohhuiying1989@gmail.com
TAY WEI GUO	tayweiguo1989@msn.com
PENG JIAYUAN	pengjiayuan2011@hotmail.com
HUANG ZHANPENG	huangzhanpeng1992@msn.com
…	

Selecting Rows

Find the names and emails of computer science students.

The WHERE clause is used for conditions.

```
SELECT name, email
FROM student
WHERE department='CS';
```

name	email
LIU SHAOJUN	liushaojun2010@msn.com
QIN YIYANG	qinyiyang2010@hotmail.com
SIOW CAO KHOA	siowcaokhoa1991@msn.com
DAVID CHAPMAN	davidchapman1989@msn.com
…	

Selecting Rows

Find the names and emails of students who graduated after '2014-01-01'.
There are several comparison operators for each domain.

```
SELECT name, email, graduate
FROM student
WHERE graduate>= '2014-01-01';
```

The condition must be TRUE (not FALSE, not UNKNOWN)

name	email	graduate
HUANG WENXIN	huangwenxin2010@msn.com	1/1/2014
NG ANH QUANG	nganhquang1991@yahoo.com	8/1/2014
QIN YIYANG	qinyiyang2010@hotmail.com	8/1/2014

Selecting Rows

Find the names and emails of students who graduated after '2014-01-01'. Null values need a special careful treatment.

```
SELECT name, email, graduate
FROM student
WHERE graduate>= '2014-01-01' OR graduate iS NULL;
```

The condition must be TRUE (not FALSE, not UNKNOWN)

name	email	graduate
PENG JIAYUAN	pengjiayuan2011@hotmail.com	
HUANG ZHANPENG	huangzhanpeng1992@msn.com	
ZHENG ZHEMIN	zhengzhemin1991@yahoo.com	
LIU ZHANPENG	liuzhanpeng2011@msn.com	
HUANG WENXIN	huangwenxin2010@msn.com	1/1/2014
WANG NA	wangna1990@yahoo.com	
NG ANH QUANG	nganhquang1991@yahoo.com	8/1/2014
…		

Querying Multiple Tables

Find the names of students and the titles of the books they own.

We can list all the tables we want to use in the FROM clause.

We are querying the Cartesian product of these tables.

```
SELECT student.name, book.title
FROM student, copy, book
WHERE student.email=copy.owner
AND copy.book=book.ISBN13;
```

name	title
DAVID HALL	The Digital Photography Book
GOH HUI YING	Photoshop Elements 9: The Missing Manual
HUANG ZHANPENG	Where Good Ideas Come From: The Natural History of Innovation
JENNY HUNT	The Great Gatsby
JENNY HUNT	The Great Gatsby
JENNY HUNT	Photoshop Elements 9: The Missing Manual
LIU SHAOJUN	The Great Gatsby
…	

Tuple Variables

Find the names of students and the titles of the books they own.
It is best to use t-uple variables (see t-uple calculus).

```
SELECT s.name, b.title
FROM student s, copy c, book b
WHERE s.email=c.owner
AND c.book=b.ISBN13;
```

name	title
DAVID HALL	The Digital Photography Book
GOH HUI YING	Photoshop Elements 9: The Missing Manual
HUANG ZHANPENG	Where Good Ideas Come From: The Natural History of Innovation
JENNY HUNT	The Great Gatsby
JENNY HUNT	The Great Gatsby
JENNY HUNT	Photoshop Elements 9: The Missing Manual
LIU SHAOJUN	The Great Gatsby
…	

Renaming

Find the names of students who lent a book returned after 2010-03-04 to anniechapman1991@yahoo.com.

Renaming of the columns in the result is done with ``AS''.

```
SELECT s.name AS owner
FROM loan l, student s
WHERE  s.email=l.owner
   AND l.returned  > '2010-03-04'
   AND l.borrower = 'anniechapman1991@yahoo.com';
```

owner
JENNY HUNT
TSO HUI LIN
JENNY HUNT
TSO HUI LIN
JENNY HUNT

Duplicates

Find the different names of students who lent a book returned after 2010-03-04 to anniechapman1991@yahoo.com.

One can eliminate duplicates rows in the result table using ``DISTINCT''.

```
SELECT DISTINCT s.name AS owner
FROM loan l, student s
WHERE  s.email=l.owner
   AND l.returned  > '2010-03-04'
   AND l.borrower = 'anniechapman1991@yahoo.com';
```

owner
JENNY HUNT
TSO HUI LIN

Ordering Rows

Find the names of students who lent a book returned after 2010-03-04 to anniechapman1991@yahoo.com in descending alpha-numerical order.

The result can be ordered using the ORDER BY clause.

```
SELECT s.name AS owner
FROM loan l, student s
WHERE s.email=l.owner
   AND l.returned  > '2010-03-04'
   AND l.borrower = 'anniechapman1991@yahoo.com'
   ORDER BY name DESC;
```

owner
TSO HUI LIN
TSO HUI LIN
JENNY HUNT
JENNY HUNT
JENNY HUNT

Ordering Rows

Find the ISBN14 of the books that have been borrowed by anniechapman1991@yahoo.com , their borrowing and return dates in ascending order of the borrowing and return dates.

We can order according to several columns.

We can order according to columns not in the result.

```
SELECT l.book, l.returned
FROM loan l
WHERE l.borrower='anniechapman1991@yahoo.com'
ORDER BY l.borrowed, l.returned;
```

book	returned
978-1449389673	6/13/2010
978-0684801520	8/13/2010
978-1449389673	8/13/2010
978-0684801520	8/13/2011
978-1449389673	8/13/2011

Arithmetic and Renaming

Find the price of books after tax (18%).

Arithmetic and other operations are available for most domains.

```
CREATE TABLE catalog (
book ISBN13 CHAR(14) PRIMARY KEY,
price number);

SELECT * FROM catalog;
```

ISBN13	price
978-0321474049	24
978-0684801520	35
978-1449389673	12
978-1594487712	55

```
SELECT ISBN13, price * 1.18 AS priceGST FROM catalog;
```

ISBN13	priceGST
978-0321474049	28.32
978-0684801520	41.3
978-1449389673	14.16
978-1594487712	64.9

Aggregate Queries: Counting Rows

Find the total number of (different) books.

Aggregate queries use aggregate functions like ``COUNT()'' to combine results over entire tables or columns.

```
SELECT COUNT(*) FROM book b;
```

COUNT(*)
4

`COUNT(), MAX(), MIN(), AVG(), STD(), SUM()` etc.

Aggregate Queries: Counting Rows

Find the total number of titles.

```
SELECT COUNT(l.book)
FROM loan l;

SELECT COUNT(ALL l.book)
FROM loan l;
```

COUNT(l.book)
56

COUNT(ALL l.book)
56

Aggregate Queries: Counting Rows

Find the total number of different titles.

```
SELECT COUNT(DISTINCT l.book)
FROM loan l;
```

COUNT(DISTINCT l.book)
4

Aggregate Queries: Average, Minimum, etc.

Find the average price of a book.

```
SELECT AVG(c.price)
FROM catalog c;
```

AVG(c.price)
31.5

Aggregate Queries: Grouping

Find, for each day, the number of books borrowed by anniechapman1991@yahoo.com.

The GROUP BY cluase creates groups before the aggregation.

```
SELECT COUNT(l.book)
FROM loan l
WHERE l.borrower='anniechapman1991@yahoo.com'
GROUP BY l.borrowed;
```

Can we get 0?

COUNT(l.book)
1
2
2

Aggregate Queries: Grouping

Which one eliminates duplicates?

```
SELECT DISTINCT l.book
FROM loan l;
```

```
SELECT l.book
FROM loan l
GROUP BY l.book;
```

Interestingly, the GROUP BY clause can be used to eleiminate duplicates. Sometimes it is the only way to do so.

For readability of the queries, unless impossible, prefer ``DISTINCT''.

Aggregate Queries: Grouping

Find, for each day, the number of books borrowed by anniechapman1991@yahoo.com, print the day and the quantity.

Only attributes in the GROUP BY clause and aggregate functions can be used in the SELECT (and HAVING) clauses.

```
SELECT l.borrowed, COUNT(l.book)
FROM loan l
WHERE l.borrower='anniechapman1991@yahoo.com'
GROUP BY l.borrowed;
```

borrowed	COUNT(l.book)
17/05/2010	1
17/07/2010	2
17/07/2011	2

Aggregate Queries: Grouping

Find, for each day, the number of books borrowed print the borrower, the day and the quantity.

```
SELECT l.borrower, l.borrowed, COUNT(l.book)
FROM loan l
GROUP BY l.borrowed;
```

Most systems: "not a GROUP BY expression“.

SQLite Ok but random.

Aggregate Queries: Grouping

Find, for each day, the number of books borrowed by anniechapman1991@yahoo.com print the borrower, the day and the quantity.

```
SELECT l.borrower, l.borrowed, COUNT(l.book)
FROM loan l
GROUP BY l.borrowed;
```

Most systems: "not a GROUP BY expression“.

SQLite Ok.

Aggregate Queries: Grouping

Find, for each day, the number of books borrowed by anniechapman1991@yahoo.com print the borrower, the day and the quantity.

```
SELECT l.borrower, l.borrowed, COUNT(l.book)
FROM loan l
WHERE l.borrower='anniechapman1991@yahoo.com'
GROUP BY l.borrowed, l.borrower;
```

borrower	borrowed	COUNT(l.book)
anniechapman1991 @yahoo.com	5/17/2010	1/1/1900
anniechapman1991 @yahoo.com	7/17/2010	1/2/1900
anniechapman1991 @yahoo.com	7/17/2011	1/2/1900

Aggregate Queries: Grouping

Find, for each day, the number of books borrowed, print the borrower, the day and the quantity.

```
SELECT l.borrower, l.borrowed, COUNT(l.book)
FROM loan l
GROUP BY l.borrowed, l.borrower;
```

borrower	borrowed	COUNT(l.book)
zhengnana1991@gmail.com	1/1/2010	1
anniechapman1991@yahoo.com	7/17/2010	2
zhouxialin1990@yahoo.com	7/17/2010	1
jennyhunt1991@gmail.com	7/20/2010	1
zhuchang2010@gmail.com	1/7/2010	1

Aggregate Queries: Grouping

What does this query find?

```
SELECT l.borrower, l.borrowed, COUNT(l.book)
FROM loan l
GROUP BY l.borrowed, l.borrower, l.book;
```

What does this query find?

```
SELECT l.borrower, l.borrowed, COUNT(l.book)
FROM loan l;
```

What does this query find?

```
SELECT COUNT(l.book) % with DISTINCT?
FROM loan l;
```

Aggregate Queries: Condition

Find the students who borrowed more that one book on any given day.
Aggregate functions cannot be used in the WHERE clause.

```
SELECT l.borrower
FROM loan l
GROUP BY l.borrowed, l.borrower
WHERE COUNT(l.book) >1;
```

"Incorrect syntax near the keyword 'WHERE'."

Aggregate Queries: Condition

Find the students who borrowed more that one book on any given day.
We need the HAVING clause.

```
SELECT l.borrower
FROM loan l
GROUP BY l.borrowed, l.borrower
HAVING COUNT(l.book) >1;
```

borrower
davidhall1992@yahoo.com
anniechapman1991@yahoo.com
anniechapman1991@yahoo.com

Aggregate Queries: Condition

Find the different students who borrowed more that one book on any given day.

```
SELECT DISTINCT l.borrower
FROM loan l
GROUP BY l.borrowed, l.borrower
HAVING COUNT(l.book) >1;
```

borrower
davidhall1992@yahoo.com
anniechapman1991@yahoo.com

Nested Queries

Find the names of the students from whom anniechapman1991@yahoo.com borrowed a book and returned it after 2010-03-04.

There can be nested queries using ``IN'', ``=ANY'', ``>ALL'' etc.

```
SELECT s.name
FROM student s
WHERE email IN (SELECT l.owner
   FROM loan l
   WHERE l.returned  > '2010-03-04'
   AND l.borrower = 'anniechapman1991@yahoo.com');
```

name
JENNY HUNT
TSO HUI LIN

Nested Queries

Find the names of the students from whom anniechapman1991@yahoo.com borrowed a book and returned it after 2010-03-04.

```
SELECT s.name
FROM student s
WHERE s.email = ANY (SELECT l.owner
   FROM loan l
   WHERE l.returned  > '2010-03-04'
   AND l.borrower = 'anniechapman1991@yahoo.com');
```

No ``ANY'' in SQLite

Nested Queries

Find the names of the students from whom anniechapman1991@yahoo.com borrowed a book and returned it after 2010-03-04.

Nested queries can sometimes be rewritten as simple queries. However, the simple query may behave slightly differently with respect to duplicates.

```
SELECT s.name
FROM loan l, student s
WHERE s.email=l.owner
   AND l.returned  > '2010-03-04'
   AND l.borrower = 'anniechapman1991@yahoo.com';
```

name
JENNY HUNT
TSO HUI LIN
JENNY HUNT
TSO HUI LIN
JENNY HUNT

Nested Queries

Find the names of the students from whom anniechapman1991@yahoo.com borrowed a book and returned it after 2010-03-04.

Nested queries can sometimes be rewritten as simple queries.

```
SELECT DISTINCT s.name
FROM loan l, student s
WHERE s.email=l.owner
   AND l.returned  > '2010-03-04'
   AND l.borrower = 'anniechapman1991@yahoo.com';
```

name
JENNY HUNT
TSO HUI LIN

Nested Queries

Find the different students from whom anniechapman1991@yahoo.com never borrowed.

The nested queries using ``NOT'' may not be rewritten into simple queries.

They add expressive power.

```
SELECT s.email
FROM student s
WHERE s.email NOT IN (SELECT l.owner
   FROM loan l
   WHERE l.borrower = 'anniechapman1991@yahoo.com');
```

email
angjiayi1990@hotmail.com
anniechapman1991@yahoo.com
chnghuiling1992@gmail.com
choyyiting1992@hotmail.com
davidchapman1989@msn.com
davidhall1992@yahoo.com
dennisbeckham1989@msn.com
…

Nested Queries

Find the different students from whom anniechapman1991@yahoo.com never borrowed.

The nested queries using ``<>ALL'' may not be rewritten into simple queries.

They add expressive power.

```
SELECT s.email
FROM student s
WHERE s.email  <> ALL (SELECT l.owner
   FROM  loan l
   WHERE l.borrower ='anniechapman1991@yahoo.com');
```

No ``ALL'' in SQLite

Nested Queries: EXISTS

Is there a book more expensive than 30$ in the catalog?

```
SELECT 'YES'
FROM catalog c
WHERE EXISTS (SELECT *
   FROM catalog c
   WHERE c.price > 30);
```

YES
YES
YES
YES
YES

```
SELECT 'YES'
FROM catalog c
WHERE EXISTS (SELECT *
   FROM catalog c
   WHERE c.price > 100);
```

No result

Nested Queries

Find the different students from whom anniechapman1991@yahoo.com never borrowed

The nested queries using ``NOT EXISTS'' may not be rewritten into simple queries.

They add expressive power.

```
SELECT s.email
FROM student s
WHERE NOT EXISTS (SELECT l.owner
   FROM loan l
   WHERE s.email = l.owner
   AND l.borrower = 'anniechapman1991@yahoo.com');
```

email
angjiayi1990@hotmail.com
anniechapman1991@yahoo.com
chnghuiling1992@gmail.com
choyyiting1992@hotmail.com
davidchapman1989@msn.com
davidhall1992@yahoo.com
…

Nested Queries (Scope)

- An attribute of an outer-query can only be used within the SELECT and WHERE clauses of the query in which its relation is declared (FROM clause) and within inner-queries (subqueries) queries.

- Attributes of the inner-queries cannot be used in the outer-queries

Nested Queries

- There can be multiple nested queries and multiple levels of nested queries

- Nested queries can appear in the WHERE but also the HAVING clauses

Simple amd nested Queries

You will learn more about simple and nested queries in the Logic and Calculus lectures.

Union

Find all the information about students in the computer science department or in the information systems department.

```
SELECT *
FROM student T
WHERE T.department='CS'
UNION
SELECT *
FROM student T
WHERE T.department='IS';
```

name	email	year	faculty	department	graduate
ANG JIA YI	angjiayi1990@hotmail.com	8/1/2009	School of Computing	CS	
DAVID CHAPMAN	davidchapman1989@msn.com	1/1/2008	School of Computing	CS	
DENNIS BECKHAM	dennisbeckham1989@msn.com	8/1/2010	School of Computing	IS	
DING WEI XIANG	dingweixiang1990@yahoo.com	1/1/2010	School of Computing	IS	
HUANG XUANTI	huangxuanti1992@msn.com	8/1/2007	School of Computing	IS	
IRIS BROWN	irisbrown1992@hotmail.com	8/1/2008	School of Computing	CS	
…					

Intersection

Find the emails of students in the computer science department owning a book with ISBN14 ‘978-0684801520’.

```
SELECT T1.email
FROM student T1
WHERE T1.department='CS'
INTERSECT
SELECT T2.owner AS email
FROM copy T2
WHERE T2.book='978-0684801520';
```

email
liushaojun2010@msn.com

Can you write the same query without INTERSECT?

(Non-Symmetric) Difference

Find the mails of students in the computer science department except those owning a book with ISBN14 ‘978-0684801520’.

```
SELECT T1.email
FROM student T1
WHERE T1.department='CS'
EXCEPT
SELECT T2.owner AS email
FROM copy T2
WHERE T2.book='978-0684801520';
```

T1.email
angjiayi1990@hotmail.com
davidchapman1989@msn.com
irisbrown1992@hotmail.com
liuzhencai1990@msn.com
ngookaiting1991@yahoo.com
ngyanfen2010@msn.com
ngyongming2011@yahoo.com
qinyiyang2010@hotmail.com
qinyuwei2011@hotmail.com
siowcaokhoa1991@msn.com

Oracle uses ``MINUS’’

Join

Find the emails of students owning a book with ISBN14 ‘978-0262033848’.

```
SELECT T1.email
FROM student T1, copy T2
WHERE T2.owner=T1.email
AND T2.book='978-0684801520';
```

email
jennyhunt1991@gmail.com
jennyhunt1991@gmail.com
liushaojun2010@msn.com
tayweiguo1989@msn.com

Inner Join

Find the emails of students owning a book with ISBN14 ‘978-0262033848’

```
SELECT T1.email
FROM student T1 INNER JOIN copy T2
ON T2.owner=T1.email
WHERE T2.book='978-0684801520';
```

email
jennyhunt1991@gmail.com
jennyhunt1991@gmail.com
liushaojun2010@msn.com
tayweiguo1989@msn.com

Left Outer Join

Find the names of the students and the titles of the books they own. If a student does not own any book, print a NULL value.

```
SELECT T1.name,T2.book
FROM student T1, copy T2
WHERE T1.email=T2.owner
UNION
SELECT T3.name, CAST(NULL AS CHAR(14)) AS book
FROM student T3
WHERE NOT EXISTS (SELECT * FROM copy T4
                  WHERE T3.email=T4.owner);
```

T1.name	T2.book
ANG JIA YI	
ANNIE CHAPMAN	
CHNG HUI LING	
CHOY YI TING	
DAVID CHAPMAN	
DAVID HALL	978-0321474049
DENNIS BECKHAM	
…	

Left Outer Join

Find the names of the students and the different titles of the books they own. If a student does not own any book, print a NULL value.

```
Select DISTINCT T1.name, T2.book
FROM student T1 LEFT OUTER JOIN copy T2
ON T1.email=T2.owner;
```

name	book
GOH HUI YING	978-1449389673
TAY WEI GUO	978-0684801520
PENG JIAYUAN	
HUANG ZHANPENG	978-1594487712
ZHENG ZHEMIN	
…	

Right Outer Join

Find the title of books and the emails of their owner. If a book does not have an owner , print a NULL value.

```
Select T2.title, T1.owner
FROM copy T1 RIGHT OUTER JOIN book T2
ON T1.book=T2.ISBN14;
```

Error: RIGHT and FULL OUTER JOINs are not currently supported

Full Outer Join

A full outer join will pad both the left and right relations with null values.

```
Select DISTINCT T2.a, T1.c
FROM table1 T1 FULL OUTER JOIN table T2
ON T1.b=T2.b
```

Other Join

- (EQUI) JOIN
- NATURAL JOIN USING
- CROSS JOIN

You will learn more about UNION, INTERSECT, EXCEPT and JOINs in the Relational Algebra lecture.

Summary

1. FROM
2. WHERE
3. GROUP BY
4. HAVING
5. ORDER BY
6. SELECT

Credits

The content of this lecture is based on chapter 5 of the book "Introduction to database Systems"
By
S. Bressan and B. Catania, McGraw Hill publisher

Clipart and media are licensed from Microsoft Office Online Clipart and Media