Өгөгдлийн сангийн үндэс (CSII202 - 3 кр)
Database Systems

Lecture 5: SQL SELECT

МУИС, ХШУИС, МКУТ-ийн багш

Док. Довдонгийн Энхзол

Пифагор /МЭӨ V зуун /

Бусдыг үгийг сонсоод дуугүй байна гэдэг ухаан сууж буйн шинж. Мэргэн ухааны дээд нь нам гүмд оршидог.

Нэн түрүүнд ухаантай байхыг чармай! Харин завтай цагтаа эрдэмтэн болно биз.

Аливаа юмны эхлэл гэдэг уг ажлын тэн хагас гэсэн үг.

In this Lecture

- SQL SELECT
  - WHERE clauses
  - SELECT from multiple tables
  - JOINs
- For more information
  - Connolly and Begg Chapter 5
  - Ullman and Widom Chapter 6.1-6.3

SQL SELECT Overview

```
SELECT
   [DISTINCT | ALL] <column-list>
  FROM <table-names>
   [WHERE <condition>]
   [ORDER BY <column-list>]
   [GROUP BY <column-list>]
   [HAVING <condition>]
```

- *([]- optional, | - or)*

Example Tables

Student

ID	First	Last
S103	John	Smith
S104	Mary	Jones
S105	Jane	Brown
S106	Mark	Jones
S107	John	Brown

Course

Code	Title
DBS	Database Systems
PR1Pro	gramming 1
PR2Pro	gramming 2
IAI Intro	to AI

Grade

ID	Code	Mark
S103	DBS	72
S103	IAI 58	
S104	PR168	
S104	IAI 65	
S106	PR243	
S107	PR176	
S107	PR260	
S107	IAI 35	

DISTINCT and ALL

- Sometimes you end up with duplicate entries
- Using **`DISTINCT`** removes duplicates
- Using **`ALL`** retains them - this is the default

```
SELECT ALL Last
  FROM Student
```

Last
Smith
Jones
Brown
Jones
Brown

```
SELECT DISTINCT Last
  FROM Student
```

Last
Smith
Jones
Brown

WHERE Clauses

- Usually you don’t want all the rows
  - A **`WHERE`** clause restricts the rows that are returned
  - It takes the form of a condition - only those rows that satisfy the condition are returned

- Example conditions:
  - **`Mark < 40`**
  - **`First = ‘John’`**
  - **`First <> ‘John’`**
  - **`First = Last`**
  - **`(First = ‘John’) AND`**

    `(Last = ‘Smith’)`
  - **`(Mark < 40) OR (Mark > 70)`**

WHERE Examples

```
SELECT * FROM Grade
 WHERE Mark >= 60
```

ID	Code	Mark
S103	DBS	72
S104	PR168	
S104	IAI 65	
S107	PR176	
S107	PR260	

```
SELECT DISTINCT ID
  FROM Grade
  WHERE Mark >= 60
```

ID
S103
S104
S107

WHERE Example

- Given the table

Grade

ID	Code	Mark
S103	DBS	72
S103	IAI 58	
S104	PR168	
S104	IAI 65	
S106	PR243	
S107	PR176	
S107	PR260	
S107	IAI 35	

- Write an SQL query to find a list of the ID numbers and marks in IAI of students who have passed (scored 40 or higher) IAI

ID	Mark
S103	58
S104	65

One Solution

We only want the ID and Mark, not the Code

Single quotes around the string

```
SELECT ID, Mark FROM Grade
  WHERE (Code = 'IAI') AND
        (Mark >= 40)
```

We're only interested in IAI

We're looking for entries with pass marks

SELECT from Multiple Tables

- Often you need to combine information from two or more tables
- You can get the effect of a product by using

```
SELECT * FROM Table1,
  Table2...
```

- If the tables have columns with the same name ambiguity results
- You resolve this by referencing columns with the table name

```
TableName.Column
```

SELECT from Multiple Tables

```
SELECT
  First, Last, Mark
FROM Student, Grade
WHERE
  (Student.ID =
     Grade.ID) AND
  (Mark >= 40)
```

Student

ID	First	Last
S103	John	Smith
S104	Mary	Jones
S105	Jane	Brown
S106	Mark	
S107	John	

Grade

ID	Code	Mark
S103	DBS	72
S103	IAI 58	
S104	PR168	
S104	IAI 65	
S106	PR243	
S107	PR176	
S107	PR260	
S107	IAI 35	

SELECT from Multiple Tables

```
SELECT ... FROM Student, Grade WHERE...
```

ID	First	Last	ID	Code	Mark
S103	John	Smith	S103	DBS	72
S103	John	Smith	S103	IAI	58
S103	John	Smith	S104	PR1	68
S103	John	Smith	S104	IAI	65
S103	John	Smith	S106	PR2	43
S103	John	Smith	S107	PR1	76
S103	John	Smith	S107	PR2	60
S103	John	Smith	S107	IAI	35
S104	Mary	Jones	S103	DBS	72
S104	Mary	Jones	S103	IAI	58
S104	Mary	Jones	S104	PR1	68
S104	Mary	Jones	S104	IAI	65
S10	Mary	Jones			

Are matched with the first entry from the Student table...

All of the entries from the Grade table

And then with the second…

and so on

SELECT from Multiple Tables

```
SELECT ... FROM Student, Grade
  WHERE (Student.ID = Grade.ID) AND ...
```

	First	Last	ID	Code	Mark
	John	Smith	S103	DBS	72
	John	Smith	S103	IAI	58
	Mary	Jones	S104	PR1	68
	Mary	Jones	S104	IAI	65
	Mark	Jones	S106	PR2	43
	John	Brown	S107	PR1	76
	John	Brown	S107	PR2	60
	John	Brown	S107	IAI	35

Student.ID Grade.ID

SELECT from Multiple Tables

```
SELECT ... FROM Student, Grade
  WHERE (Student.ID = Grade.ID) AND (Mark >= 40)
```

ID	First	Last	ID	Code	Mark
S103	John	Smith	S103	DBS	72
S103	John	Smith	S103	IAI	58
S104	Mary	Jones	S104	PR1	68
S104	Mary	Jones	S104	IAI	65
S106	Mark	Jones	S106	PR2	43
S107	John	Brown	S107	PR1	76
S107	John	Brown	S107	PR2	60

SELECT from Multiple Tables

```
SELECT First, Last, Mark FROM Student, Grade
  WHERE (Student.ID = Grade.ID) AND (Mark >= 40)
```

First	Last	Mark
John	Smith	72
John	Smith	58
Mary	Jones	68
Mary	Jones	65
Mark	Jones	43
John	Brown	76
John	Brown	60

SELECT from Multiple Tables

- When selecting from multiple tables you almost always use a **WHERE** clause to find entries with common values

```
SELECT * FROM
  Student, Grade,
  Course
 WHERE
  Student.ID = Grade.ID
 AND
  Course.Code =
  Grade.Code
```

SELECT from Multiple Tables

Student			Grade			Course	
ID	First	Last	ID	Code	Mark	Code	Title
S103	John	Smith	S103	DBS	72	DBS	Database Systems
S103	John	Smith	S103	IAI	58	IAI	Intro to AI
S104	Mary	Jones	S104	PR1	68	PR1	Programming 1
S104	Mary	Jones	S104	IAI	65	IAI	Intro to AI
S106	Mark	Jones	S106	PR2	43	PR2	Programming 2
S107	John	Brown	S107	PR1	76	PR1	Programming 1
S107	John	Brown	S107	PR2	60	PR2	Programming 2
S107	John	Brown	S107	IAI	35	IAI	Intro to AI

Student.ID = Grade.ID　　　Course.Code = Grade.Code

JOINs

- JOINs can be used to combine tables
  - There are many types of JOIN
    - **`CROSS JOIN`**
    - **`INNER JOIN`**
    - **`NATURAL JOIN`**
    - **`OUTER JOIN`**
  - **`OUTER JOIN`**s are linked with **`NULL`**s - more later

`A CROSS JOIN B`

- returns all pairs of rows from A and B

`A NATURAL JOIN B`

- returns pairs of rows with common values for identically named columns and without duplicating columns

`A INNER JOIN B`

- returns pairs of rows satisfying a condition

CROSS JOIN

Student

ID	Name
123	John
124	Mary
125	Mark
126	Jane

Enrolment

ID	Code
123	DBS
124	PRG
124	DBS
126	PRG

```
SELECT * FROM
  Student CROSS JOIN
  Enrolment
```

ID	Name	ID	Code
123	John	123	DBS
124	Mary	123	DBS
125	Mark	123	DBS
126	Jane	123	DBS
123	John	124	PRG
124	Mary	124	PRG
125	Mark	124	PRG
126	Jane	124	PRG
123	John	124	DBS
124	Mar		DBS

NATURAL JOIN

Student

ID	Name
123	John
124	Mary
125	Mark
126	Jane

Enrolment

ID	Code
123	DBS
124	PRG
124	DBS
126	PRG

```
SELECT * FROM
  Student NATURAL JOIN
  Enrolment
```

ID	Name	Code
123	John	DBS
124	Mary	PRG
124	Mary	DBS
126	Jane	PRG

CROSS and NATURAL JOIN

```
SELECT * FROM
  A CROSS JOIN B
```

- is the same as

```
SELECT * FROM A, B
```

```
SELECT * FROM
  A NATURAL JOIN B
```

- is the same as

```
SELECT A.col1,… A.coln,
[and all other columns
apart from B.col1,…B.coln]
FROM A, B
WHERE A.col1 = B.col1
  AND A.col2 = B.col2
...AND A.coln = B.col.n
(this assumes that col1…
coln in A and B have
common names)
```

INNER JOIN

- **`INNER JOIN`**s specify a condition which the pairs of rows satisfy

```
SELECT * FROM
   A INNER JOIN B
   ON <condition>
```

- Can also use

```
SELECT * FROM
   A INNER JOIN B
   USING
     (col1, col2,…)
```

- Chooses rows where the given columns are equal

INNER JOIN

Student

ID	Name
123	John
124	Mary
125	Mark
126	Jane

Enrolment

ID	Code
123	DBS
124	PRG
124	DBS
126	PRG

```
SELECT * FROM
   Student INNER JOIN
   Enrolment USING (ID)
```

ID	Name	ID	Code
123	John	123	DBS
124	Mary	124	PRG
124	Mary	124	DBS
126	Jane	126	PRG

INNER JOIN

Buyer

Name	Budget
Smith	100,000
Jones	150,000
Green	80,000

Property

Address Pr	ice
15 High St	85,000
12 Queen St 12	5,000
87 Oak Row 17	5,000

```
SELECT * FROM
  Buyer INNER JOIN
  Property ON
  Price <= Budget
```

Name	Budget	Address	Price
Smith	100,000	15 High St	85,000
Jones	150,000	15 High St	85,000
Jones	150,000	12 Queen St	125,000

INNER JOIN

```
SELECT * FROM
   A INNER JOIN B
   ON <condition>
```

- is the same as

```
SELECT * FROM A, B
   WHERE <condition>
```

```
SELECT * FROM
   A INNER JOIN B
 USING(col1, col2,...)
```

- is the same as

```
SELECT * FROM A, B
 WHERE A.col1 = B.col1
   AND A.col2 = B.col2
   AND ...
```

JOINs vs WHERE Clauses

- JOINs (so far) are not needed
  - You can have the same effect by selecting from multiple tables with an appropriate WHERE clause
  - So should you use JOINs or not?
- Yes, because
  - They often lead to concise queries
  - NATURAL JOINs are very common
- No, because
  - Support for JOINs varies a fair bit among SQL dialects

Writing Queries

- When writing queries
  - There are often many ways to write the query
  - You should worry about being correct, clear, and concise in that order
  - Don’t worry about being clever or efficient

- Most DBMSs have query optimisers
  - These take a user’s query and figure out how to efficiently execute it
  - A simple query is easier to optimise
  - We’ll look at some ways to improve efficiency later