Medical Informatics

Lecture 6: Introduction to SQL

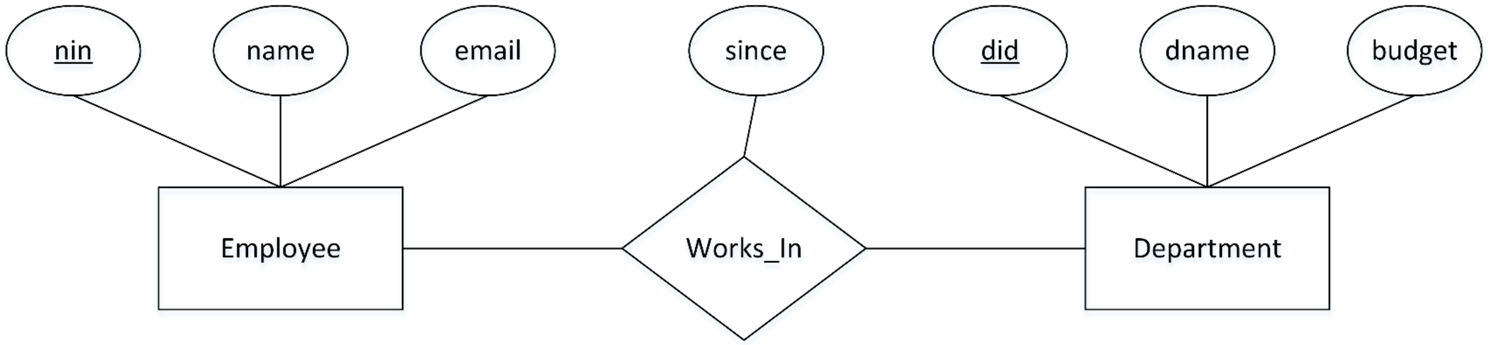
Dr Areti Manataki

Nanjing Medical University

In the previous lecture

- Systematically transform an ER model into a relational one
- Transforming:
  - entity and relationship sets
  - key and participation constraints
  - weak entity sets and hierarchies

In the previous lecture



```
CREATE TABLE Employee (
    nin CHAR(9),
    name VARCHAR(20),
    email VARCHAR(35),
    PRIMARY KEY (nin) )
CREATE TABLE Department (
    did INTEGER,
    dname VARCHAR(20),
    budget INTEGER,
    PRIMARY KEY (did) )
```

```
CREATE TABLE Works_In (
    nin CHAR(9),
    did INTEGER,
    since INTEGER,
    PRIMARY KEY (nin, did),
    FOREIGN KEY (nin) REFERENCES
              Employee,
    FOREIGN KEY (did) REFERENCES
              Department )
```

In this lecture

- We’ll learn how to use the SQL Data Manipulation Language to
  - insert, delete and update rows in a table
  - query the database

Inserting rows into a table

```
CREATE TABLE Student (
    mn CHAR(8),
    name CHAR(20),
    email CHAR(25),
    age INTEGER,
    PRIMARY KEY (mn) )
```

```
INSERT
  INTO Student (mn, name, email, age)
  VALUES (‘s1253477’, ‘Jenny’, ‘jenny@sms.ed.ac.uk’, 23)
```

- The above statement adds a tuple in the Student table.
- We could omit the list of column names and simply list the values in the appropriate order, but it is good practice to include column names.

Deleting and updating rows

- We can delete tuples using the DELETE command

```
DELETE
    FROM Student
    WHERE name = ‘Alan’
```

- We can update the column values in an existing row using the UPDATE command

```
UPDATE Student
    SET name = ‘Alan’
    WHERE mn = ‘s1428571’
```

SQL queries

- SQL allows us to ask questions to the database, such as:
  - Which students are older than 19?
  - What are the names of all students taking the Medical Informatics course?
  - What is the average age of all students born in Europe who are taking the Medical Informatics course but not the Advanced Databases course?

A simple SQL query

- The following query returns all students older than 19.

```
SELECT *
FROM Student
WHERE age > 19
```

- The * means that the table returned has the same schema as Students.

mn	name	email	age
s0785212	Andrew	andrew@maths	19
s1253477	Jenny	jenny@inf	23
s1456381	Rhona	rhona@med	18
s1489673	Stuart	stuart@med	34
s1473612	Alan	alan@law	23

A simple SQL query

- The following query returns all students older than 19.

```
SELECT *
FROM Student
WHERE age > 19
```

- The * means that the table returned has the same schema as Students.

mn	name	email	age
s0785212	Andrew	andrew@maths	19
s1253477	Jenny	jenny@inf	23
s1456381	Rhona	rhona@med	18
s1489673	Stuart	stuart@med	34
s1473612	Alan	alan@law	23

A simple SQL query

- The following query returns all students older than 19.

```
SELECT *
FROM Student
WHERE age > 19
```

- The * means that the table returned has the same schema as Students.

mn	name	email	age
s1253477	Jenny	jenny@inf	23
s1489673	Stuart	stuart@med	34
s1473612	Alan	alan@law	23

SQL query syntax

```
SELECT [DISTINCT] field-list
FROM table-list
[ WHERE qualification ]
```

- Anything in [*square brackets*] is optional.
- SELECT: the columns to be retained in the result
- FROM: the tables from which to take the data
- WHERE: conditions that should hold for the records to be picked out

Variations of a simple SQL query

- Instead of using *, we can explicitly specify the list of fields to be returned. These could be in a different order than in the original table.

```
SELECT *
FROM Student
WHERE age > 19
```

```
SELECT mn, name, email, age
FROM Student
WHERE age > 19
```

Variations of a simple SQL query

- We can specify which tables the fields are from.
- This is particularly useful when the FROM-clause includes several tables.

```
SELECT *
FROM Student
WHERE age > 19
```

```
SELECT Student.mn, Student.name,
       Student.email, Student.age
FROM Student
WHERE Student.age > 19
```

Variations of a simple SQL query

- We can specify which tables the fields are from, while locally abbreviating their names.
- This is particularly useful when the FROM-clause includes several tables.

```
SELECT *           SELECT S.mn, S.name, S.email,
FROM Student                      S.age
WHERE age > 19     FROM Student S
                   WHERE S.age > 19
```

Additional SQL queries

- We may choose to select only a subset of the fields of each selected tuple.

```
SELECT S.name
FROM Student S
WHERE S.age > 19
```

- In this case, the table returned has a different schema to that in Student.

name
Jenny
Stuart
Alan

Additional SQL queries

- We may choose not to specify a condition through the WHERE-part of the query.

```
SELECT age
FROM Student
```

age
19
23
18
34
23

- By using DISTINCT, we remove any duplicates from the returned records.

```
SELECT DISTINCT age
FROM Student
```

age
19
23
18
34

Additional SQL queries

- We can include several tables in the FROM-clause.
- The following query returns the email addresses of all students taking Medical Informatics.

```
SELECT S.email
FROM Student S, Takes T, Course C
WHERE S.mn = T.mn
   AND T.cid = C.cid
   AND C.title = ‘Medical Informatics’
```

Query evaluation

```
SELECT S.email
FROM Student S, Takes T, Course C
WHERE S.mn = T.mn AND T.cid = C.cid
   AND C.title = ‘Medical Informatics’
```

1. Take all rows from the tables.

mn	name	email	age
s0785212	Andrew	andrew@maths	19
s1253477	Jenny	jenny@inf	23
s1456381	Rhona	rhona@med	18
s1489673	Stuart	stuart@med	34
s1473612	Alan	alan@law	23

mn	cid
s0785212	lalg
s1253477	dbs
s1253477	medinf
s1489673	medinf
s1473612	sls

cid	title	credits
dbs	Database Systems	20
inf1	Informatics 1	10
medinf	Medical Informatics	10
sls	Scottish Legal System	10
lalg	Linear Algebra	10

Query evaluation

```
SELECT S.email
FROM Student S, Takes T, Course C
WHERE S.mn = T.mn AND T.cid = C.cid
   AND C.title = ‘Medical Informatics’
```

1. Take all rows from the tables.
2. Keep only the row combinations that satisfy the qualification conditions.

mn	name	email	age
s0785212	Andrew	andrew@maths	19
s1253477	Jenny	jenny@inf	23
s1456381	Rhona	rhona@med	18
s1489673	Stuart	stuart@med	34
s1473612	Alan	alan@law	23

mn	cid
s0785212	lalg
s1253477	dbs
s1253477	medinf
s1489673	medinf
s1473612	sls

cid	title	credits
dbs	Database Systems	20
inf1	Informatics 1	10
medinf	Medical Informatics	10
sls	Scottish Legal System	10
lalg	Linear Algebra	10

Query evaluation

```
SELECT S.email
FROM Student S, Takes T, Course C
WHERE S.mn = T.mn AND T.cid = C.cid
   AND C.title = ‘Medical Informatics’
```

1. Take all rows from the tables.
2. Keep only the row combinations that satisfy the qualification conditions.

mn	name	email	age
s0785212	Andrew	andrew@maths	19
s1253477	Jenny	jenny@inf	23
s1456381	Rhona	rhona@med	18
s1489673	Stuart	stuart@med	34
s1473612	Alan	alan@law	23

mn	cid
s0785212	lalg
s1253477	dbs
s1253477	medinf
s1489673	medinf
s1473612	sls

cid	title	credits
dbs	Database Systems	20
inf1	Informatics 1	10
medinf	Medical Informatics	10
sls	Scottish Legal System	10
lalg	Linear Algebra	10

Query evaluation

```
SELECT S.email
FROM Student S, Takes T, Course C
WHERE S.mn = T.mn AND T.cid = C.cid
   AND C.title = ‘Medical Informatics’
```

1. Take all rows from the tables.
2. Keep only the row combinations that satisfy the qualification conditions.
3. Return the specified columns.

mn	name	email	age
s0785212	Andrew	andrew@maths	19
s1253477	Jenny	jenny@inf	23
s1456381	Rhona	rhona@med	18
s1489673	Stuart	stuart@med	34
s1473612	Alan	alan@law	23

mn	cid
s0785212	lalg
s1253477	dbs
s1253477	medinf
s1489673	medinf
s1473612	sls

cid	title	credits
dbs	Database Systems	20
inf1	Informatics 1	10
medinf	Medical Informatics	10
sls	Scottish Legal System	10
lalg	Linear Algebra	10

Query evaluation

```
SELECT S.email
FROM Student S, Takes T, Course C
WHERE S.mn = T.mn AND T.cid = C.cid
   AND C.title = ‘Medical Informatics’
```

1. Take all rows from the tables.
2. Keep only the row combinations that satisfy the qualification conditions.
3. Return the specified columns.

email
jenny@inf
stuart@med

Conclusions

- We’ve been introduced to the SQL Data Manipulation Language to:
  - insert, delete and update rows in a table
  - query the database
- General form of a basic SQL query:

```
SELECT [DISTINCT] field-list
FROM table-list
[ WHERE qualification ]
```

- In the next lecture we’ll learn how to formulate more complex queries in SQL.

Acknowledgements

The content of these slides was originally created for the Medical Informatics course from The University of Edinburgh, which is licensed under a Creative Commons Attribution-ShareAlike 4.0 International (CC BY-SA 4.0) license.