

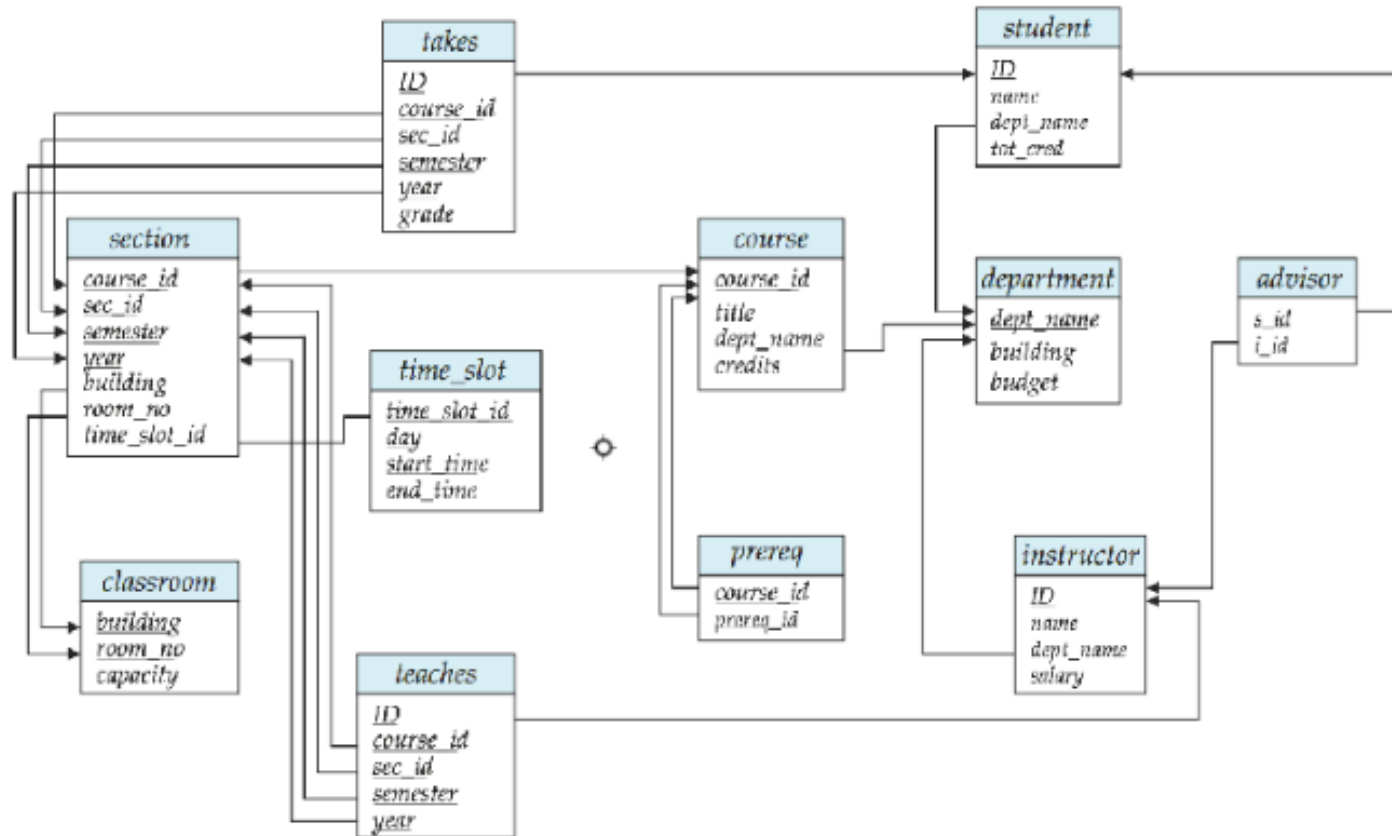
Multiple Tables

CMPSC 305 – Database Systems



ALLEGHENY COLLEGE

How to Connect Information? - No free-standing tables allowed!



The Basic Query Structure

The SQL data-manipulation language (DML) provides the ability to query information, and insert, delete and update tuples

A typical SQL code has the form:

```
SELECT A1, A2, ..., An  
FROM r1, r2, ..., rm  
WHERE P;
```

- A_i represents an attribute
- R_i represents a relation
- P is a predicate
- The result of an SQL query is a relation

The SELECT Clause

The SELECT clause filters out particular data from a table.

- SQL allows duplicates in relations as well as in query results.
- The SELECT statement has many optional clauses:
 - **WHERE** specifies which rows to retrieve.
 - **GROUP BY** groups rows sharing a property so that an aggregate function can be applied to each group.
 - **HAVING** selects among the groups defined by the GROUP BY clause.
 - **ORDER BY** specifies an order in which to return the rows.
 - **AS** provides an alias which can be used to temporarily rename tables or columns.

Given table 'T'

Table "T"	Query	Result												
<table><tr><th>C1</th><th>C2</th></tr><tr><td>1</td><td>a</td></tr><tr><td>2</td><td>b</td></tr></table>	C1	C2	1	a	2	b	<pre>SELECT * FROM T;</pre>	<table><tr><th>C1</th><th>C2</th></tr><tr><td>1</td><td>a</td></tr><tr><td>2</td><td>b</td></tr></table>	C1	C2	1	a	2	b
C1	C2													
1	a													
2	b													
C1	C2													
1	a													
2	b													
<table><tr><th>C1</th><th>C2</th></tr><tr><td>1</td><td>a</td></tr><tr><td>2</td><td>b</td></tr></table>	C1	C2	1	a	2	b	<pre>SELECT C1 FROM T;</pre>	<table><tr><th>C1</th></tr><tr><td>1</td></tr><tr><td>2</td></tr></table>	C1	1	2			
C1	C2													
1	a													
2	b													
C1														
1														
2														
<table><tr><th>C1</th><th>C2</th></tr><tr><td>1</td><td>a</td></tr><tr><td>2</td><td>b</td></tr></table>	C1	C2	1	a	2	b	<pre>SELECT * FROM T WHERE C1 = 1;</pre>	<table><tr><th>C1</th><th>C2</th></tr><tr><td>1</td><td>a</td></tr></table>	C1	C2	1	a		
C1	C2													
1	a													
2	b													
C1	C2													
1	a													
<table><tr><th>C1</th><th>C2</th></tr><tr><td>1</td><td>a</td></tr><tr><td>2</td><td>b</td></tr></table>	C1	C2	1	a	2	b	<pre>SELECT * FROM T ORDER BY C1 DESC;</pre>	<table><tr><th>C1</th><th>C2</th></tr><tr><td>2</td><td>b</td></tr><tr><td>1</td><td>a</td></tr></table>	C1	C2	2	b	1	a
C1	C2													
1	a													
2	b													
C1	C2													
2	b													
1	a													

The TeaDB

Build file: sandbox/teaDB/teaDB Build.txt

- `cat builder_teaTableDB.txt | sqlite3 teaDB.sqlite3`

(or just copy in the text into SQLite3!)

The SELECT Clause

Find everything in the Department table.

- `SELECT * FROM Department;`

Find all entries for dept's of the Department table

- `SELECT dept from Department;`

Count entries of dept's in Department table,

- `SELECT COUNT(dept) FROM department;`

The SELECT Clause

Find all unique entries for depts in Department table,

- `SELECT DISTINCT(dept) FROM department;`

Count unique entries of depts in Department table,

- `SELECT COUNT(DISTINCT(dept)) FROM Department;`

Return an exhaustive set of sandwiches that are being ordered.

- `SELECT DISTINCT(sandwich) FROM Tea;`

What query to use to count these types of sandwiches?