Query Language
- The language we use to “talk” to a database
- Statements – “sentences” of a query language
  - Queries
  - Updates

Selection
- Most commonly used statement
- Retrieve from the database some records that satisfy certain conditions

General form:
SELECT field_name(s)
FROM table_name(s)
WHERE condition(s);

Some SQL Conventions
- All capital letters for SQL keywords
- Capitalize first letter of table names
- All lower-case letters for field names

No Conditions
- WHERE clause can be omitted

Display all movie titles:
SELECT title FROM Box_Office;

Selecting Field(s)
- Select single field
- Select multiple field
  - * – all fields

Display all movies titles with their weekend grosses:
SELECT title, ?? FROM Box_Office;

Display all movie titles and their box office rankings for this weekend:
SELECT title, this_week FROM Box_Office;

Display all box office information:
SELECT * FROM Box_Office;
**Column Aliases**
- Change column headings of the query results
  - Better readability

Display all movies titles with their weekend grosses:

```
SELECT title AS [Movie Title], wgross AS [Weekend Gross]
FROM Box_Office;
```

**Ordering the Results**
- ORDER BY field_name(s)
- Ascending and descending order
  - ASC
  - DESC

Display all movies titles with their weekend grosses and cumulative grosses:
- unordered results
- order by weekend grosses in descending order
- order by cumulative grosses in ascending order
- both

**Conditions**

```
SELECT field_name(s) FROM table_name(s) WHERE condition(s);
```

- Predicate(s)

Display all movies which have grossed more than 100M so far:

```
SELECT title AS [Movie Title], cgross AS [Cumulative Gross]
FROM Box_Office
WHERE cgross > 100,000,000;
```

**Comparison Operators**
- Attribute types that can be compared:
  - Numerical
  - Currency
  - Text
  - Date and Time

- Generally speaking, different attribute types cannot compare to each other
- Greater than: >
- Greater than or equal to: >=
- Less than: <
- Less than or equal to: <=

**Multiple Conditions**

- Combine multiple conditions with AND and OR

Display all movies which have grossed more than 1M this weekend and more than 100M after it's released:

```
SELECT title, wgross, cgross
FROM Box_Office
WHERE wgross > 1,000,000 AND cgross > 100,000,000;
```

- Combine two conditions with BETWEEN

```
f BETWEEN a AND b ⇔ f >= a AND f <= b
```

Display the movies which ranked 10 to 20 during this weekend:

```
SELECT title, this_week
FROM Box_Office
WHERE this_week BETWEEN 10 AND 20;
```

How about BETWEEN 20 AND 10??
Multiple Conditions

- Combine two conditions with NOT BETWEEN
- NOT in general

NULL

- When the information is missing or unknown
- NULL has no data type, and cannot be compared to other values

Find all students who have not selected a major yet:

```sql
SELECT name FROM Student WHERE major IS NULL;
```

Find all students who have already selected a major

Find all movies which are released this week

String Matching Using LIKE

```sql
f LIKE pattern
```

- Pattern
  - Zero or more characters: *
  - Any single character: ?
  - Any single digit: #

Find all movies which has the word “love” in its title:

```sql
SELECT title FROM Box_office WHERE title LIKE "*love*";
```

Combine Information from Different Fields

- Find the weekend gross per theater of each movie
- ORDER BY

Remove Repeated Results

```sql
SELECT DISTINCT f FROM ... WHERE ...;
```

- Find the departments which provide courses that are more than three credit hours

COUNT Function

- The number of results rather than the results themselves

How many movies were released during this weekend?

```sql
SELECT COUNT(title) FROM Box_office WHERE week=1;
```
Working with Multiple Tables

- Find the class room capacity of class section 85
- Table Qualifier

Table Aliases

- Find the class room capacity of class section 85

```sql
SELECT r.capacity
FROM Section s, Room r
WHERE s = 85 AND s.bldg = r.bldg AND s.room = r.room;
```

Subqueries

- Find the average grosses of the top 10 movie during this week end.
- Note that the result of a SQL is still a *table*