CS186 Discussion 2

(single-table SQL, multi-table SQL)

Matthew Deng

Single-Table SQL

Single-Table SQL

```
SELECT [DISTINCT] <column expression list>
FROM <single table>
[WHERE <predicate>]
[GROUP BY <column list>
 [HAVING <predicate>]]
[ORDER BY <column list>];
```

SELECT FROM

```
SELECT [DISTINCT] <column expression list>
FROM <single table>
[WHERE <predicate>]
[GROUP BY <column list>
 [HAVING <predicate>]]
[ORDER BY <column list>];
```

- Retrieve entries from the table in the FROM clause
- Output columns in the SELECT clause

WHERE

```
SELECT [DISTINCT] <column expression list>
FROM <single table>
[WHERE <predicate>]
[GROUP BY <column list>
 [HAVING <predicate>]]
[ORDER BY <column list>];
```

- Keep only the tuples that satisfy the predicate in the WHERE clause

DISTINCT

```
SELECT [DISTINCT] <column expression list>
FROM <single table>
[WHERE <predicate>]
[GROUP BY <column list>
 [HAVING <predicate>]]
[ORDER BY <column list>];
```

- Remove duplicate tuples before outputting

GROUP BY

```
SELECT [DISTINCT] <column expression list>
FROM <single table>
[WHERE <predicate>]
[GROUP BY <column list>
 [HAVING <predicate>]]
[ORDER BY <column list>];
```

- Partition table into groups in GROUP BY predicate
- Produces aggregate result for each group
- Aggregates: AVG, SUM, COUNT, MAX, MIN

HAVING

```
SELECT [DISTINCT] <column expression list>
FROM <single table>
[WHERE <predicate>]
[GROUP BY <column list>
 [HAVING <predicate>]]
[ORDER BY <column list>];
```

- Keep only the tuples that satisfy the predicate in the HAVING clause
- Can be used on aggregates or GROUP BY columns
- Can ONLY be used with GROUP BY

ORDER BY

```
SELECT [DISTINCT] <column expression list>
FROM <single table>
[WHERE <predicate>]
[GROUP BY <column list>
 [HAVING <predicate>]]
[ORDER BY <column list>];
```

- Sorts results by columns from left to right
  - Students.age, Students.name
- ASC for ascending [default], DESC for descending
- Must refer to columns in output

Other Clauses

- LIKE
  - Compare similar values with wildcard operators
  - In WHERE clause
  - ‘_’: exactly one character
  - ‘%’: any number of characters
- LIMIT
  - At the end of query
  - Number of results to return

Worksheet #1-4

Single-Table SQL Exercises

1. Find the 5 songs that spent the most weeks in the top 40, ordered from most to least.

Single-Table SQL Exercises

1. Find the 5 songs that spent the most weeks in the top 40, ordered from most to least.

```
SELECT song_name
FROM Songs
ORDER BY weeks_in_top_40
DESC LIMIT 5;
```

Single-Table SQL Exercises

2. Find the name and first year active of every artist whose name starts with the letter ‘B’.

Single-Table SQL Exercises

2. Find the name and first year active of every artist whose name starts with the letter ‘B’.

```
SELECT artist_name, first_year_active
FROM Artists
WHERE artist_name LIKE ‘B%’;
```

Single-Table SQL Exercises

3. Find the total number of “Techno” albums released each year.

Single-Table SQL Exercises

3. Find the total number of “Techno” albums released each year.

```
SELECT year_released, COUNT(*)
FROM Albums
WHERE genre = ‘Techno’
GROUP BY year_released;
```

Single-Table SQL Exercises

4. Find the genre and the number of albums released per genre; don’t include genres that have a count of less than 10.

Single-Table SQL Exercises

4. Find the genre and the number of albums released per genre; don’t include genres that have a count of less than 10.

```
SELECT genre, COUNT(*)
FROM Albums
GROUP BY genre
HAVING COUNT(*) >= 10;
```

Multi-Table SQL

Single-Table SQL

```
SELECT [DISTINCT] <column expression list>
FROM <single table>
[WHERE <predicate>]
[GROUP BY <column list>
 [HAVING <predicate>]]
[ORDER BY <column list>];
```

Multi-Table SQL

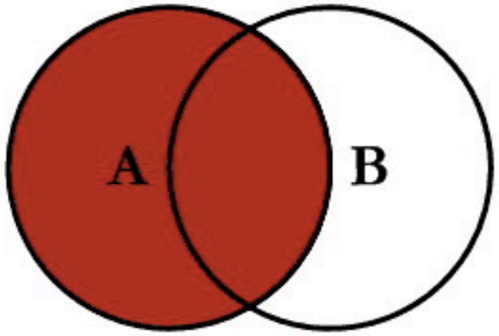
```
SELECT [DISTINCT] <column expression list>
FROM <multiple tables>
[WHERE <predicate>]
[GROUP BY <column list>
 [HAVING <predicate>]]
[ORDER BY <column list>];
```

Multi-Table SQL

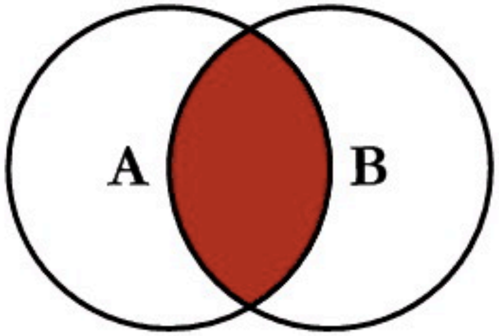
- Multiple tables in the FROM clause
- Join predicate in the WHERE clause
  - Boats.owner_id = Sailors.sid

- FROM <table 1> INNER JOIN <table 2> ON <predicate>

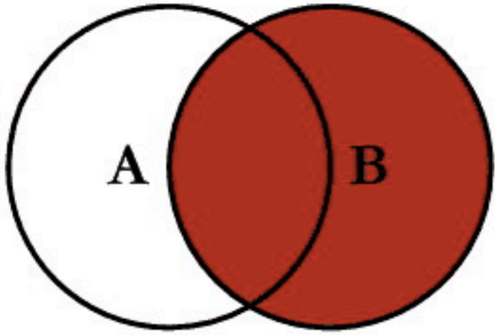
SQL JOINS



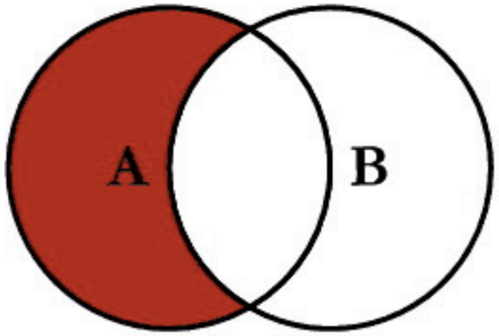
```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
```



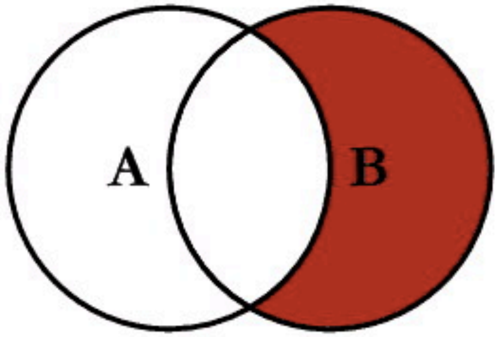
```
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
```



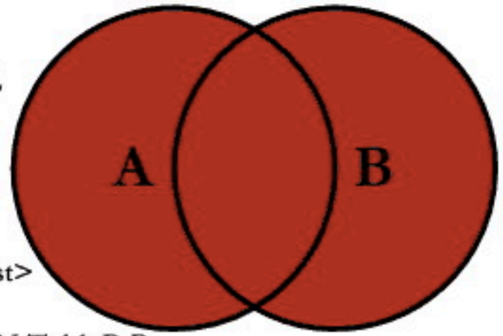
```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
```



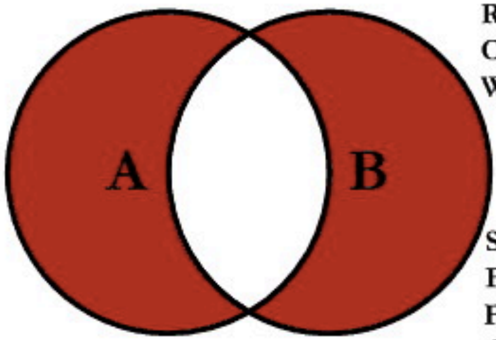
```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
```



```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
```



```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
```



```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
OR B.Key IS NULL
```

Worksheet #5-8

Multi-Table SQL Exercises

5. The name of all songs with the genre “Country” which have spent more than 2 weeks in the top 40.

Multi-Table SQL Exercises

5. The name of all songs with the genre “Country” which have spent more than 2 weeks in the top 40.

```
SELECT Songs.song_name
FROM Albums, Songs
WHERE Songs.album_id = Albums.album_id
    AND Albums.genre = 'country’
    AND Songs.weeks_in_top_40 > 2;
```

Multi-Table SQL Exercises

6. For each song, its name, the name of its album, and the name of its artist.

Multi-Table SQL Exercises

6. For each song, its name, the name of its album, and the name of its artist.

```
SELECT Songs.song_name,
       Albums.album_name,
       Artists.artist_name
FROM Artists, Albums, Songs
WHERE Artists.artist_id = Albums.artist_id
AND Songs.album_id = Albums.album_id;
```

Multi-Table SQL Exercises

7. The artist name and number of albums released by each artist.

Multi-Table SQL Exercises

7. The artist name and number of albums released by each artist.

```
SELECT Artists.artist_name, COUNT(*)
FROM Artists, Albums
WHERE Artists.artist_id = Albums.artist_id
GROUP BY Artists.artist_id,
         Artists.artist_name;
```

Multi-Table SQL Exercises

8. Find singers, with no duplicates, who released both “Techno” and “Pop” albums.

Multi-Table SQL Exercises

8. Find singers, with no duplicates, who released both “Techno” and “Pop” albums.

```
SELECT DISTINCT Artists.artist_name
FROM Artists, Albums A1, Albums A2
WHERE Artists.artist_id = A1.artist_id
  AND A1.artist_id = A2.artist_id
  AND A1.genre = ‘Techno’
  AND A2.genre = ‘Pop’;
```