

CSE 344 Section 2 Worksheet

1. Joins Examples

Given tables created with these commands:

```
CREATE TABLE A (a int);
CREATE TABLE B (b int);
INSERT INTO A VALUES (1), (2), (3), (4);
INSERT INTO B VALUES (3), (4), (5), (6);
```

What's the output for each of the following:

```
SELECT *
  FROM A INNER JOIN B
    ON A.a=B.b;
```

```
a|b
3|3
4|4
```

```
SELECT *
  FROM A RIGHT OUTER JOIN B
    ON A.a=B.b;
```

```
a|b
3|3
4|4
  |5
  |6
```

```
SELECT *
  FROM A LEFT OUTER JOIN B
    ON A.a=B.b;
```

```
a|b
1|
2|
3|3
4|4
```

```
SELECT *
  FROM A FULL OUTER JOIN B
    ON A.a=B.b;
```

```
a|b
1|
2|
3|3
4|4
  |5
  |6
```

SELECT * FROM A INNER JOIN B; (Challenging question!)

```
a|b
1|3
1|4
1|5
1|6
2|3
2|4
2|5
2|6
3|3
3|4
3|5
3|6
4|3
4|4
4|5
4|6
```

Sidenote:

sqlite3 supports neither RIGHT OUTER nor FULL OUTER.

Right outer can be implemented with

```
SELECT *
  FROM B LEFT OUTER JOIN A
    ON A.a=B.b;
```

Full outer can be implemented with

```
SELECT *
  FROM A LEFT OUTER JOIN B
    ON A.a=B.b
UNION
SELECT *
  FROM B LEFT OUTER JOIN A
    ON A.a=B.b;
```

2. Self Join

Consider the following over simplified Employee table:

```
CREATE TABLE Employees (id int, bossId int);
```

Suppose all employees have an id which is not null. How would we find all distinct pairs of employees with the same boss?

```
SELECT E1.id, E2.id
FROM Employees AS E1, Employees AS E2
WHERE E1.id < E2.id
      AND E1.bossId = E2.bossId;
```

Sidenote: The predicate "E1.id < E2.id" could also be written as "E1.id > E2.id". We cannot use plain inequality "E1.id <> E2.id" as the predicate condition because this would lead to duplicate pairs.

3. 3 Valued-Logic

Given the table created with these commands:

```
CREATE TABLE A (a int, b int);
INSERT INTO A VALUES (1, 1), (2, 10), (3, NULL);
```

What is the output for each of the following:

```
SELECT A.a FROM A
WHERE A.b < 5;
```

1

```
SELECT A.a FROM A
WHERE A.b >= 5;
```

2

```
SELECT A.a FROM A
WHERE A.b != 1 AND
      A.b != 10;
```

[no output]
(3, NULL) returns unknown for both conditions due to 3-value logic.

```
SELECT A.a FROM A
WHERE A.b < 5 OR
      A.b IS NULL;
```

1
3
(3, NULL) returns true on the second condition only