Assignment 3

1)

SQL:

```
SELECT NAME
FROM climbed, participated
WHERE climbed.PEAK = 'Pilot Knob (S)' AND
      climbed.TRIP_ID = participated.TRIP_ID;
```

OUTPUT:

NAME

JOHN
MARK
MICHAEL

2)

SQL:

```
SELECT PEAK
FROM participated, climbed
WHERE participated.NAME = 'MARK' AND
      participated.TRIP_ID = climbed.TRIP_ID;
```

OUTPUT:

PEAK

North Maggie Mountain
Whaleback
Center Peak
Mount Langley
Mount Hale
Whaleback
Midway Mountain
Kearsarge Peak
Pilot Knob (S)
Lion Rock
South Guard
Mount Langley
Dragon Peak
Mount Barnard
Mount Newcomb
Thor Peak
Mount Guyot
North Guard
Mount Langley
Mount McAdie
Muah Mountain
Mount Rixford
Mount Guyot

Olancha Peak
Whaleback
Kearsarge Peak
Lion Rock
Joe Devel Peak
Angora Mountain
Florence Peak
Needham Mountain
Midway Mountain
Mount Williamson
Moses Mountain

3)
SQL:

```
SELECT NAME
FROM participated
WHERE TRIP_ID =
      (SELECT TRIP_ID
      FROM peak, climbed
      WHERE peak.DIFF = 5 AND
            peak.NAME = climbed.PEAK);
```

OUTPUT:

NAME
JOHN
ELIZABETH
DONNA

4)
SQL:

```
SELECT PEAK, COUNT(WHEN)
FROM climbed
GROUP BY PEAK
HAVING COUNT(WHEN) = 1;
```

OUTPUT:

PEAK	**COUNT(WHEN)**
Mount Bradley	1
Spanish Needle	1
Angora Mountain	1
Kern Peak	1
Owens Peak	1
Crag Peak	1
Mount Whitney	1
Cartago Peak	1
Coyote Peaks	1

```
University Peak         1
Mount Gardiner          1
North Maggie Mountain   1
Pilot Knob (S)          1
Dragon Peak             1
Mount Genevra           1
Triple Divide Peak      1
Mount Muir              1
Mount Pickering         1
Center Peak             1
Lamont Peak             1
Mount Stanford (S)      1
Mount Clarance King     1
Vandever Mountain       1
Mount LeConte           1
Mount Barnard           1
```

5)

SQL:

```
SELECT DISTINCT peak.NAME
FROM climbed, participated, peak
WHERE participated.NAME = 'JOHN'              AND
      participated.TRIP_ID = climbed.TRIP_ID AND
      climbed.PEAK = peak.NAME                AND
      peak.ELEV > 14000;
```

OUTPUT:

```
NAME
Mount Whitney
Mount Langley
```

6)

SQL:

```
SELECT MAP, MAX(ELEV), MIN(ELEV)
FROM peak
GROUP BY MAP
HAVING MAX(ELEV) > MIN(ELEV) + 2000;
```

OUTPUT:

Map	**MAX(ELEV)**	**MIN(ELEV)**
Mount Whitney	14491	12300

7)

SQL:

```
SELECT MAP, AVG(ELEV)
FROM peak
```

```
GROUP BY MAP
ORDER BY AVG(ELEV);
```

OUTPUT:

MAP	AVG(ELEV)
Lament Peak	7635
Ninemile Canyon	8000
Rockhouse Basin	8360
Owens Peak	8453
Cannell Peak	8802
Silver City	9023
Crag Peak	9480
Monache Mtn	9533
Moses Mtn	9782.5
Sirretta Peak	9977
Kern Lake	10545
Bartlett	11016
Mt Silliman	11188
Lodgepole	11240
Olancha	11301.5
Kern Peak	11510
Sphinx Lakes	11717
Mineral King	12280.4
Kearsarge Peak	12427.666666666666666666666666666666667
Triple Divide Peak	12657.375
Mt Clarence King	12838.375
Cirque Peak	12900
Mt Kaweah	12945
Mt Brewer	13334.357142857142857142857142857142857
Mount Whitney	13493.277777777777777777777777777777778
Mt Langley	13561
Mt Williamson	13747.777777777777777777777777777777778

8)

SQL:

```
SELECT PEAK
FROM participated, climbed
WHERE participated.NAME = 'MARK' AND
      participated.TRIP_ID = climbed.TRIP_ID

INTERSECT

SELECT PEAK
FROM participated, climbed
WHERE participated.NAME = 'MARY' AND
      participated.TRIP_ID = climbed.TRIP_ID;
```

OUTPUT:

PEAK

Dragon Peak
Joe Devel Peak
Kearsarge Peak
Lion Rock
Midway Mountain
Moses Mountain
Mount Barnard
Mount Guyot
Mount Hale
Mount Langley
Mount McAdie
Mount Newcomb
Mount Rixford
Mount Williamson
Needham Mountain
North Guard
South Guard
Thor Peak

9)

SQL:

```
SELECT peak.REGION, COUNT(peak.NAME)
FROM peak
INNER JOIN
      (SELECT NAME
      FROM peak

      MINUS

      SELECT PEAK
      FROM climbed) diff
ON peak.NAME = diff.NAME
GROUP BY peak.REGION;
```

OUTPUT:

REGION	**COUNT(PEAK.NAME)**
Kearsarge Pass	4
Whitney to Williamson	5
Kings Kern Divide	6
Great Western Divide	3
Southern Sierra	3
Corocoran to Whitney	2
Mineral King	1

Kaweahs and West	9
Olancha to Langley	3

10)

SQL:

```
SELECT participated.TRIP_ID, SUM(ELEV)
FROM participated
INNER JOIN
      (SELECT *
      FROM climbed, peak
      WHERE climbed.PEAK = peak.NAME) elevation
ON participated.TRIP_ID = elevation.TRIP_ID
GROUP BY participated.TRIP_ID
HAVING SUM(elev) >= 500000;
```

OUTPUT:

TRIP_ID	**SUM(ELEV)**
13	564735
8	595895
3	532092

11)

SQL:

```
SELECT trips.SEX, peaksClimbed/gender
FROM
      (SELECT SEX, COUNT(PEAK) AS peaksClimbed
      FROM climbed
      INNER JOIN
            (SELECT climber.NAME, SEX, TRIP_ID
            FROM climber, participated
            WHERE climber.NAME = participated.NAME) peaks
      ON climbed.TRIP_ID = peaks.TRIP_ID
      GROUP BY SEX) trips
INNER JOIN
      (SELECT SEX, COUNT(NAME) AS gender
      FROM climber
      GROUP BY SEX) people
ON trips.SEX = people.SEX;
```

OUTPUT:

SEX	**PEAKSCLIMBED/GENDER**
M	15.76923076923076923076923076923076923077
F	21.4

12)

SQL:

```
SELECT NAME
FROM
      (SELECT DISTINCT NAME, notMaria.PEAK AS NMPEAK, Maria.PEAK AS MPEAK
      FROM
            (SELECT DISTINCT climber.NAME, participated.TRIP_ID, PEAK
            FROM climber, participated, climbed
            WHERE climber.NAME <> 'MARIA' AND
                  climber.NAME = participated.NAME AND
                  participated.TRIP_ID = climbed.TRIP_ID) notMaria
      RIGHT JOIN
            (SELECT PEAK
            FROM climber, participated, climbed
            WHERE climber.NAME = 'MARIA' AND
                  climber.NAME = participated.NAME AND
                  participated.TRIP_ID = climbed.TRIP_ID) Maria
      ON notMaria.PEAK = Maria.PEAK) matches
GROUP BY matches.NAME
HAVING COUNT(matches.NMPEAK) = 5;
```

OUTPUT:

NAME

KENNETH

PATRICIA

13)

SQL:

```
SELECT REGION, FRAC
FROM
      (SELECT pks.REGION, peaksClimbed/peaksTotal AS FRAC
      FROM
            (SELECT REGION, COUNT(NAME) AS peaksTotal
            FROM peak
            GROUP BY REGION) pks
      INNER JOIN
            (SELECT REGION, COUNT(NAME) AS peaksClimbed
            FROM
                  (SELECT DISTINCT REGION, NAME
                  FROM climbed, peak
                  WHERE climbed.PEAK = peak.NAME)
            GROUP BY REGION) cd_pk
      ON pks.REGION = cd_pk.REGION) all_reg
WHERE FRAC =
      (SELECT MAX(peaksClimbed/peaksTotal) AS MAX_FRAC
      FROM
            (SELECT REGION, COUNT(NAME) AS peaksTotal
            FROM peak
```

```
            GROUP BY REGION) pks
      INNER JOIN
            (SELECT REGION, COUNT(NAME) AS peaksClimbed
            FROM
                  (SELECT DISTINCT REGION, NAME
                  FROM climbed, peak
                  WHERE climbed.PEAK = peak.NAME)
            GROUP BY REGION) cd_pk
      ON pks.REGION = cd_pk.REGION);
```

OUTPUT:

REGION	**FRAC**
Mineral King	0.9

14)

SQL:

```
SELECT NAME1, NAME2, COUNT(PEAK) AS NUM_PEAKS
FROM climbed
INNER JOIN
      (SELECT NAME1, NAME2, cr1.TRIP_ID
      FROM
            (SELECT climber.NAME AS NAME1, TRIP_ID
            FROM climber, participated
            WHERE climber.NAME = participated.NAME) cr1
      INNER JOIN
            (SELECT climber.NAME AS NAME2, TRIP_ID
            FROM climber, participated
            WHERE climber.NAME = participated.NAME) cr2
      ON cr1.NAME1 != cr2.NAME2 AND cr1.TRIP_ID = cr2.TRIP_ID) pairs
ON climbed.TRIP_ID = pairs.TRIP_ID
GROUP BY NAME1, NAME2
HAVING COUNT(PEAK) = (
      SELECT MAX(CNT)
      FROM (
            SELECT NAME1, NAME2, COUNT(PEAK) AS CNT
            FROM climbed
            INNER JOIN
                  (SELECT NAME1, NAME2, cr1.TRIP_ID
                  FROM
                        (SELECT climber.NAME AS NAME1, TRIP_ID
                        FROM climber, participated
                        WHERE climber.NAME = participated.NAME) cr1
                  INNER JOIN
                        (SELECT climber.NAME AS NAME2, TRIP_ID
                        FROM climber, participated
                        WHERE climber.NAME = participated.NAME) cr2
```

```
                ON cr1.NAME1 != cr2.NAME2 AND cr1.TRIP_ID = cr2.TRIP_ID)
pairs
            ON climbed.TRIP_ID = pairs.TRIP_ID
            GROUP BY NAME1, NAME2));
```

OUTPUT:

NAME1	NAME2	NUM_PEAKS
KENNETH	PATRICIA	31
PATRICIA	KENNETH	31

I know this is essentially a duplicate entry, but I couldn't figure out a good way to eliminate this occuring.

15) ## This query is wrong, but I don't think it's far. I just can't figure out how to limit the date range dynamically. For example, if I replace the second parameter of the BETWEEN clause with a hard-coded date, it limits appropriately to only the people that climbed >20 peaks before then, but I'm not sure how to have a range that moves along all the dates.

SQL:

```
SELECT climber.NAME, COUNT(PEAK)
FROM climber, participated, climbed
WHERE climber.NAME = participated.NAME AND
      participated.TRIP_ID = climbed.TRIP_ID AND
      climbed.WHEN BETWEEN climbed.WHEN AND climbed.WHEN + 60
GROUP BY climber.NAME
HAVING COUNT(PEAK) > 20;
```

OUTPUT:

NAME	COUNT(PEAK)
KENNETH	51
LINDA	29
MARK	34
STEVEN	23
ELIZABETH	22
JOHN	35
DONNA	32
PATRICIA	79