Part 1

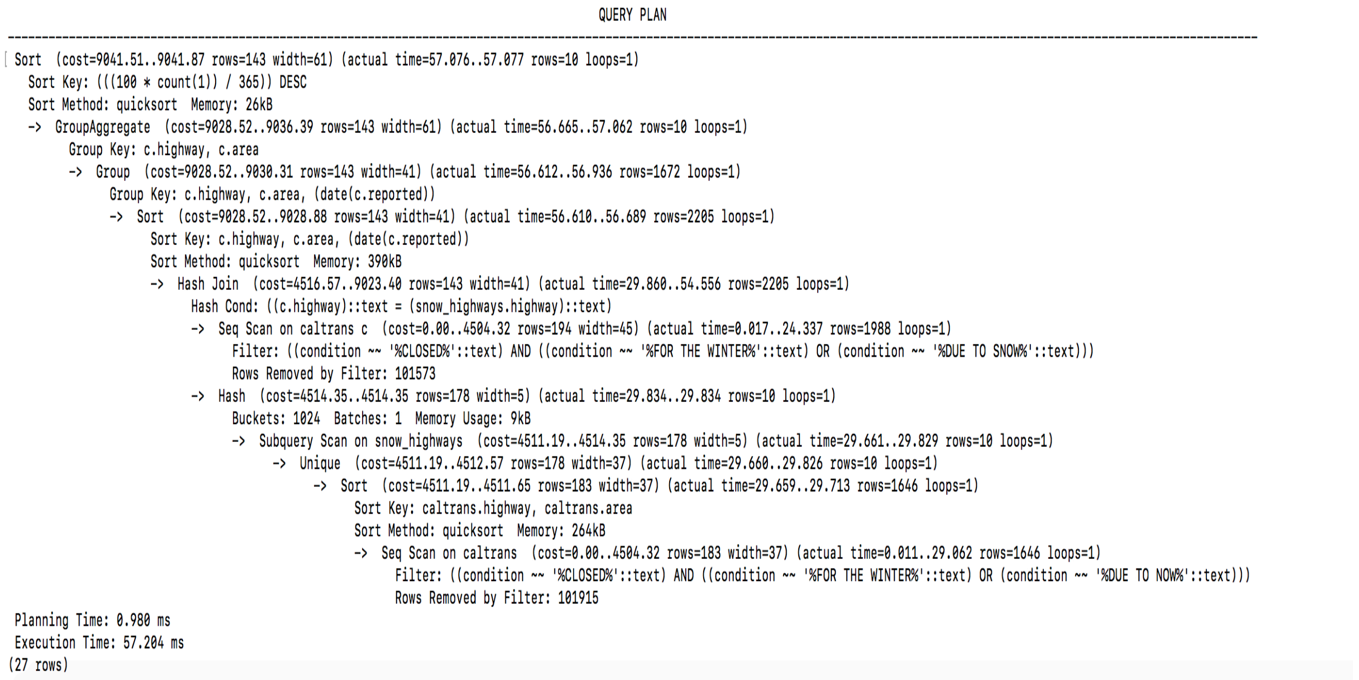
No Subquery with Distinct

```
                                                      QUERY PLAN
--------------------------------------------------------------------------------------------------------------------------------
 Sort  (cost=4500.56..4504.36 rows=1519 width=53) (actual time=46.825..46.826 rows=23 loops=1)
   Sort Key: (((count(DISTINCT date_part('doy'::text, reported)) * 100) / 365)) DESC
   Sort Method: quicksort  Memory: 28kB
   ->  GroupAggregate  (cost=4354.95..4420.29 rows=1519 width=53) (actual time=46.179..46.803 rows=23 loops=1)
         Group Key: highway, area
         ->  Sort  (cost=4354.95..4359.94 rows=1998 width=45) (actual time=46.096..46.173 rows=1988 loops=1)
               Sort Key: highway, area
               Sort Method: quicksort  Memory: 301kB
               ->  Seq Scan on caltrans  (cost=0.00..4245.41 rows=1998 width=45) (actual time=0.168..45.306 rows=1988 loops=1)
                     Filter: ((condition ~~ '%CLOSED%DUE TO SNOW%'::text) OR (condition ~~ '%CLOSED%FOR THE WINTER%'::text))
                     Rows Removed by Filter: 101573
 Planning Time: 0.643 ms
 Execution Time: 46.911 ms
(13 rows)
```

Select within a Select

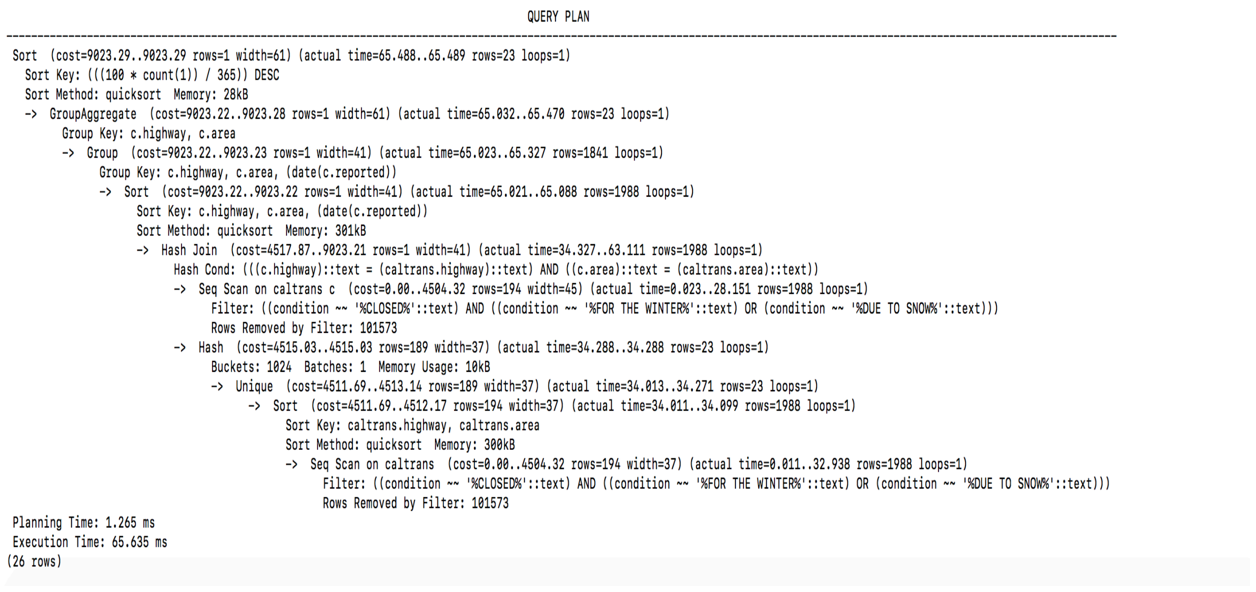
```
                                                                    QUERY PLAN
----------------------------------------------------------------------------------------------------------------------------------------
----------------
 Sort  (cost=4530.04..4530.52 rows=192 width=61) (actual time=33.012..33.013 rows=23 loops=1)
   Sort Key: (((100 * count(1)) / 365)) DESC
   Sort Method: quicksort  Memory: 28kB
   ->  GroupAggregate  (cost=4512.17..4522.75 rows=192 width=61) (actual time=32.590..32.993 rows=23 loops=1)
         Group Key: caltrans.highway, caltrans.area
         ->  Group  (cost=4512.17..4514.59 rows=192 width=41) (actual time=32.580..32.864 rows=1841 loops=1)
               Group Key: caltrans.highway, caltrans.area, (date(caltrans.reported))
               ->  Sort  (cost=4512.17..4512.66 rows=194 width=41) (actual time=32.578..32.643 rows=1988 loops=1)
                     Sort Key: caltrans.highway, caltrans.area, (date(caltrans.reported))
                     Sort Method: quicksort  Memory: 301kB
                     ->  Seq Scan on caltrans  (cost=0.00..4504.80 rows=194 width=41) (actual time=0.022..30.707 rows=1988 loops=1)
                           Filter: ((condition ~~ '%CLOSED%'::text) AND ((condition ~~ '%FOR THE WINTER%'::text) OR (condition ~~ '%DUE TO 
SNOW%'::text)))
                           Rows Removed by Filter: 101573
 Planning Time: 0.844 ms
 Execution Time: 33.964 ms
(15 rows)
```

Join as a Filter



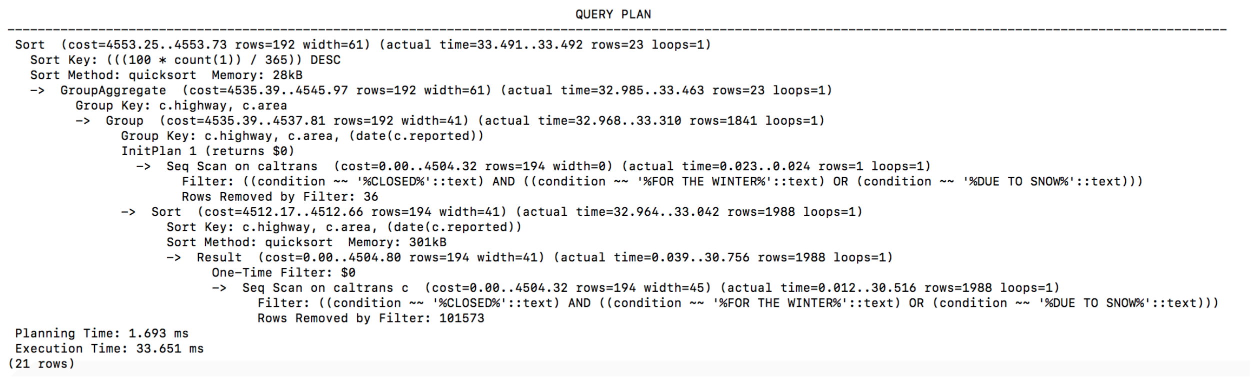
```
                                                                 QUERY PLAN
------------------------------------------------------------------------------------------------------------------------------------------------------------------------
 Sort  (cost=9041.51..9041.87 rows=143 width=61) (actual time=57.076..57.077 rows=10 loops=1)
   Sort Key: (((100 * count(1)) / 365)) DESC
   Sort Method: quicksort  Memory: 26kB
   ->  GroupAggregate  (cost=9028.52..9036.39 rows=143 width=61) (actual time=56.665..57.062 rows=10 loops=1)
         Group Key: c.highway, c.area
         ->  Group  (cost=9028.52..9030.31 rows=143 width=41) (actual time=56.612..56.936 rows=1672 loops=1)
               Group Key: c.highway, c.area, (date(c.reported))
               ->  Sort  (cost=9028.52..9028.88 rows=143 width=41) (actual time=56.610..56.689 rows=2205 loops=1)
                     Sort Key: c.highway, c.area, (date(c.reported))
                     Sort Method: quicksort  Memory: 390kB
                     ->  Hash Join  (cost=4516.57..9023.40 rows=143 width=41) (actual time=29.860..54.556 rows=2205 loops=1)
                           Hash Cond: ((c.highway)::text = (snow_highways.highway)::text)
                           ->  Seq Scan on caltrans c  (cost=0.00..4504.32 rows=194 width=45) (actual time=0.017..24.337 rows=1988 loops=1)
                                 Filter: ((condition ~~ '%CLOSED%'::text) AND ((condition ~~ '%FOR THE WINTER%'::text) OR (condition ~~ '%DUE TO SNOW%'::text)))
                                 Rows Removed by Filter: 101573
                           ->  Hash  (cost=4514.35..4514.35 rows=178 width=5) (actual time=29.834..29.834 rows=10 loops=1)
                                 Buckets: 1024  Batches: 1  Memory Usage: 9kB
                                 ->  Subquery Scan on snow_highways  (cost=4511.19..4514.35 rows=178 width=5) (actual time=29.661..29.829 rows=10 loops=1)
                                       ->  Unique  (cost=4511.19..4512.57 rows=178 width=37) (actual time=29.660..29.826 rows=10 loops=1)
                                             ->  Sort  (cost=4511.19..4511.65 rows=183 width=37) (actual time=29.659..29.713 rows=1646 loops=1)
                                                   Sort Key: caltrans.highway, caltrans.area
                                                   Sort Method: quicksort  Memory: 264kB
                                                   ->  Seq Scan on caltrans  (cost=0.00..4504.32 rows=183 width=37) (actual time=0.011..29.062 rows=1646 loops=1)
                                                         Filter: ((condition ~~ '%CLOSED%'::text) AND ((condition ~~ '%FOR THE WINTER%'::text) OR (condition ~~ '%DUE TO NOW%'::text)))
                                                         Rows Removed by Filter: 101915
 Planning Time: 0.980 ms
 Execution Time: 57.204 ms
(27 rows)
```

Using an in Subquery as a Filter



```
                                                                 QUERY PLAN
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------
 Sort  (cost=9023.29..9023.29 rows=1 width=61) (actual time=65.488..65.489 rows=23 loops=1)
   Sort Key: (((100 * count(1)) / 365)) DESC
   Sort Method: quicksort  Memory: 28kB
   ->  GroupAggregate  (cost=9023.22..9023.28 rows=1 width=61) (actual time=65.032..65.470 rows=23 loops=1)
         Group Key: c.highway, c.area
         ->  Group  (cost=9023.22..9023.23 rows=1 width=41) (actual time=65.023..65.327 rows=1841 loops=1)
               Group Key: c.highway, c.area, (date(c.reported))
               ->  Sort  (cost=9023.22..9023.22 rows=1 width=41) (actual time=65.021..65.088 rows=1988 loops=1)
                     Sort Key: c.highway, c.area, (date(c.reported))
                     Sort Method: quicksort  Memory: 301kB
                     ->  Hash Join  (cost=4517.87..9023.21 rows=1 width=41) (actual time=34.327..63.111 rows=1988 loops=1)
                           Hash Cond: (((c.highway)::text = (caltrans.highway)::text) AND ((c.area)::text = (caltrans.area)::text))
                           ->  Seq Scan on caltrans c  (cost=0.00..4504.32 rows=194 width=45) (actual time=0.023..28.151 rows=1988 loops=1)
                                 Filter: ((condition ~~ '%CLOSED%'::text) AND ((condition ~~ '%FOR THE WINTER%'::text) OR (condition ~~ '%DUE TO SNOW%'::text)))
                                 Rows Removed by Filter: 101573
                           ->  Hash  (cost=4515.03..4515.03 rows=189 width=37) (actual time=34.288..34.288 rows=23 loops=1)
                                 Buckets: 1024  Batches: 1  Memory Usage: 10kB
                                 ->  Unique  (cost=4511.69..4513.14 rows=189 width=37) (actual time=34.013..34.271 rows=23 loops=1)
                                       ->  Sort  (cost=4511.69..4512.17 rows=194 width=37) (actual time=34.011..34.099 rows=1988 loops=1)
                                             Sort Key: caltrans.highway, caltrans.area
                                             Sort Method: quicksort  Memory: 300kB
                                             ->  Seq Scan on caltrans  (cost=0.00..4504.32 rows=194 width=37) (actual time=0.011..32.938 rows=1988 loops=1)
                                                   Filter: ((condition ~~ '%CLOSED%'::text) AND ((condition ~~ '%FOR THE WINTER%'::text) OR (condition ~~ '%DUE TO SNOW%'::text)))
                                                   Rows Removed by Filter: 101573
 Planning Time: 1.265 ms
 Execution Time: 65.635 ms
(26 rows)
```

Formal Left Semijoin



```
                                                                 QUERY PLAN
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------
 Sort  (cost=4553.25..4553.73 rows=192 width=61) (actual time=33.491..33.492 rows=23 loops=1)
   Sort Key: (((100 * count(1)) / 365)) DESC
   Sort Method: quicksort  Memory: 28kB
   ->  GroupAggregate  (cost=4535.39..4545.97 rows=192 width=61) (actual time=32.985..33.463 rows=23 loops=1)
         Group Key: c.highway, c.area
         ->  Group  (cost=4535.39..4537.81 rows=192 width=41) (actual time=32.968..33.310 rows=1841 loops=1)
               Group Key: c.highway, c.area, (date(c.reported))
               InitPlan 1 (returns $0)
                 ->  Seq Scan on caltrans  (cost=0.00..4504.32 rows=194 width=0) (actual time=0.023..0.024 rows=1 loops=1)
                       Filter: ((condition ~~ '%CLOSED%'::text) AND ((condition ~~ '%FOR THE WINTER%'::text) OR (condition ~~ '%DUE TO SNOW%'::text)))
                       Rows Removed by Filter: 36
               ->  Sort  (cost=4512.17..4512.66 rows=194 width=41) (actual time=32.964..33.042 rows=1988 loops=1)
                     Sort Key: c.highway, c.area, (date(c.reported))
                     Sort Method: quicksort  Memory: 301kB
                     ->  Result  (cost=0.00..4504.80 rows=194 width=41) (actual time=0.039..30.756 rows=1988 loops=1)
                           One-Time Filter: $0
                           ->  Seq Scan on caltrans c  (cost=0.00..4504.32 rows=194 width=45) (actual time=0.012..30.516 rows=1988 loops=1)
                                 Filter: ((condition ~~ '%CLOSED%'::text) AND ((condition ~~ '%FOR THE WINTER%'::text) OR (condition ~~ '%DUE TO SNOW%'::text)))
                                 Rows Removed by Filter: 101573
 Planning Time: 1.693 ms
 Execution Time: 33.651 ms
(21 rows)
```

My definition of efficiency is based on the execution time. From the above five figures, we can see that the second and the fifth method are the fastest. The operation of hash join is time-consuming. That is why the third and the fourth one are slow. When comparing the first and the second method, we can find that the first method works on more rows than the second one, because the second method do a select first to filter some rows. This is matched with our intuition that operation on fewer rows will lead to high efficiency.

Part 2

1. Relation R will be scanned for 100,000 loops as for each row in L, R will be scanned.

2(a) Assuming there are |L| tuples in relation L, then when the index has not already been built, each tuple in both L and R will be scanned for |L| in R and once in L.

(b) Without the index precomputed on the join key:

$b_L + b_L * b_R$

with the index precomputed:
b_L*(N+J) + b_L

Part 3
1. Since A->BC, B->D, CD->E, we can know A->BCDE, and then A->ADE. A is super key of R2. R1∪R2=R, R1∩R2=A. Thus, it is lossless.

2. {A->B, C->A, C->B}

3. It is not BCNF. We can derive function dependencies, A->B, A->C, A->D, A->E. However, we could not include A->F in closure of A. To normalize it, we can decompose it into R_1(A, B,C), R_2 (B, D), R_3 (C, E), R_4 (A, F).

4. The other tuples we can know are
(a, b_1, c_1, d_2) (a, b_1, c_1, d_3) (a, b_2, c_2, d_1), (a, b_2, c_2, d_3), (a, b_3, c_3, d_1), (a, b_3, c_3, d_1)

5. It is not in 4NF. There are two MVDs, and AB is the super key. The MVD between A and B violates the restriction of 4NF. To normalize it, we can decompose it into R_1(A, B), R_2 (A, C), R_3 (A, D), R_4 (A, E), R_5 (A, F)

6. It is neither in 2NF nor in 3NF. The super key is (Venue, Year, Band). Band->Genre. However, the Genre only depends on Band. Due to the partial dependency on the key, it cannot be 2NF, and then it cannot be 3NF.

7. The super key is (Venue, Year, Band). And the function dependency is (Venue, Year, Band)-> Attendees. There is no partial dependency on the key and no transitive dependency, so it is both in 2NF and in 3NF.