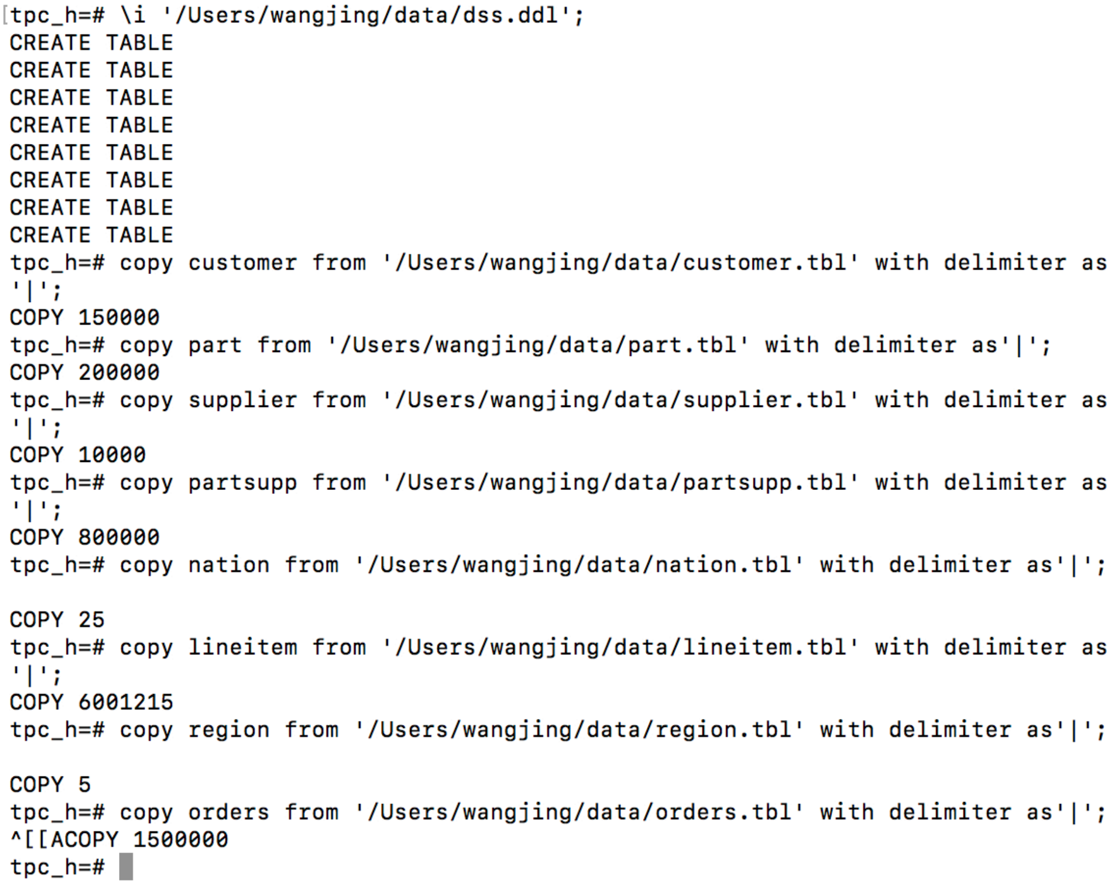
Task1:

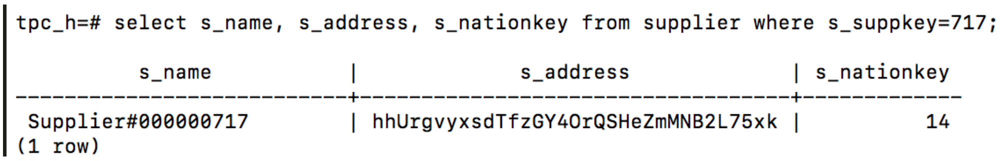
创建了一个



```
[tpc_h=# \i '/Users/wangjing/data/dss.ddl';
CREATE TABLE
CREATE TABLE
CREATE TABLE
CREATE TABLE
CREATE TABLE
CREATE TABLE
CREATE TABLE
CREATE TABLE
tpc_h=# copy customer from '/Users/wangjing/data/customer.tbl' with delimiter as
'|';
COPY 150000
tpc_h=# copy part from '/Users/wangjing/data/part.tbl' with delimiter as'|';
COPY 200000
tpc_h=# copy supplier from '/Users/wangjing/data/supplier.tbl' with delimiter as
'|';
COPY 10000
tpc_h=# copy partsupp from '/Users/wangjing/data/partsupp.tbl' with delimiter as
'|';
COPY 800000
tpc_h=# copy nation from '/Users/wangjing/data/nation.tbl' with delimiter as'|';

COPY 25
tpc_h=# copy lineitem from '/Users/wangjing/data/lineitem.tbl' with delimiter as
'|';
COPY 6001215
tpc_h=# copy region from '/Users/wangjing/data/region.tbl' with delimiter as'|';

COPY 5
tpc_h=# copy orders from '/Users/wangjing/data/orders.tbl' with delimiter as'|';
^[[ACOPY 1500000
tpc_h=# █
```

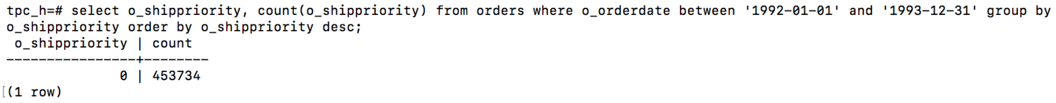
Task2:

(a) 展示 suppkey 为 717 的 supplier 的 name，address 和 nationkey，那么就要在表 supplier 中进行查找。



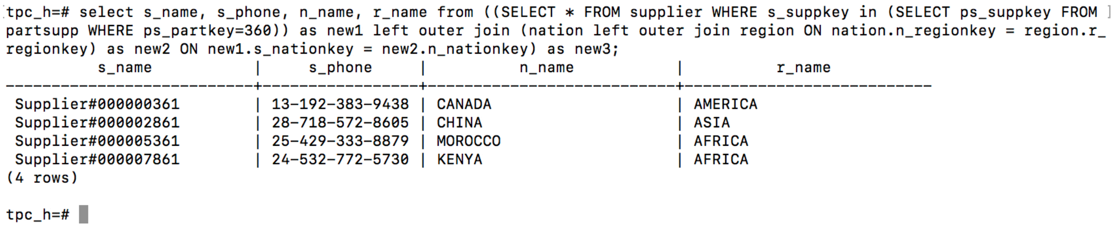
```
tpc_h=# select s_name, s_address, s_nationkey from supplier where s_suppkey=717;

          s_name           |             s_address              | s_nationkey
---------------------------+------------------------------------+-------------
 Supplier#000000717        | hhUrgvyxsdTfzGY4OrQSHeZmMNB2L75xk  |          14
(1 row)
```

(b) 对于每个 ship priority，计算其在 1992 到 1993 之间的总数，并降序排列展示。操作过程中发现，ship priority 都为 0。



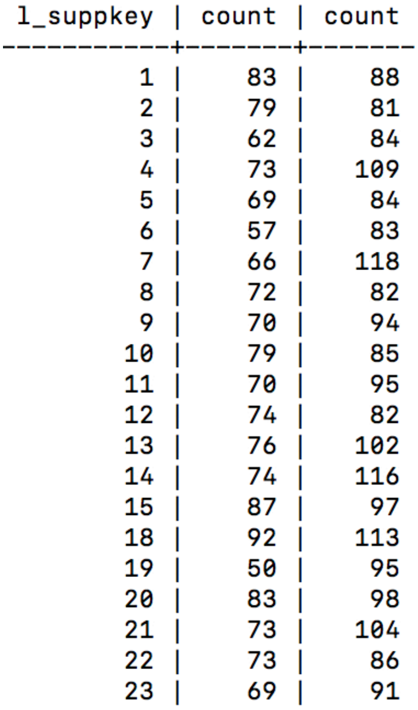
```
tpc_h=# select o_shippriority, count(o_shippriority) from orders where o_orderdate between '1992-01-01' and '1993-12-31' group by
o_shippriority order by o_shippriority desc;
 o_shippriority | count
----------------+--------
              0 | 453734
[(1 row)
```

(c) 找到为 partkey 为 part 的 supplier，然后展示它们的 name，phone，nation 和 region。



```
tpc_h=# select s_name, s_phone, n_name, r_name from ((SELECT * FROM supplier WHERE s_suppkey in (SELECT ps_suppkey FROM partsupp WHERE ps_partkey=360)) as new1 left outer join (nation left outer join region ON nation.n_regionkey = region.r_regionkey) as new2 ON new1.s_nationkey = new2.n_nationkey) as new3;
          s_name           |     s_phone     |          n_name           |          r_name
---------------------------+-----------------+---------------------------+---------------------------
 Supplier#000000361        | 13-192-383-9438 | CANADA                    | AMERICA
 Supplier#000002861        | 28-718-572-8605 | CHINA                     | ASIA
 Supplier#000005361        | 25-429-333-8879 | MOROCCO                   | AFRICA
 Supplier#000007861        | 24-532-772-5730 | KENYA                     | AFRICA
(4 rows)

tpc_h=# █
```

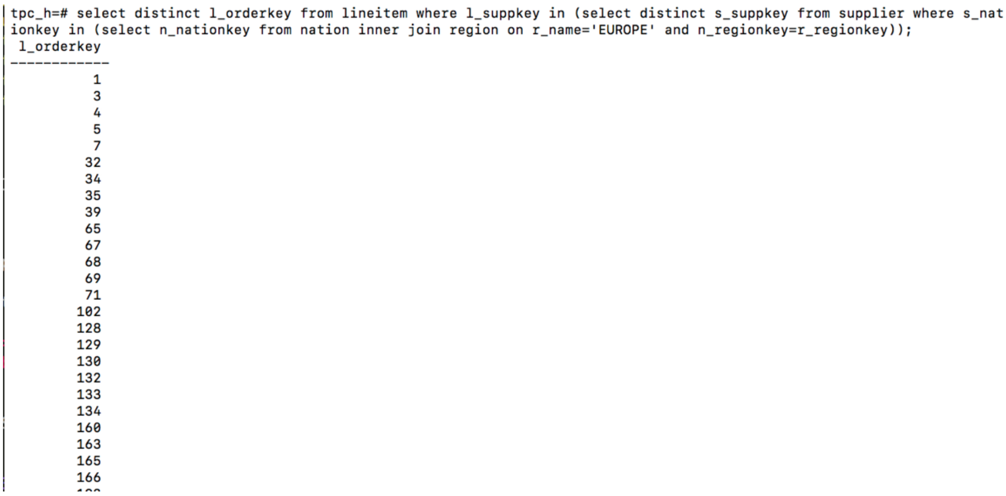
(d) 找到那些 1993 年供应比 1992 年更多 part 的 supplier，然后展示它们的及其在 1992 年和 1993 年分别的供应量。



```
 l_suppkey | count | count
-----------+-------+-------
         1 |    83 |    88
         2 |    79 |    81
         3 |    62 |    84
         4 |    73 |   109
         5 |    69 |    84
         6 |    57 |    83
         7 |    66 |   118
         8 |    72 |    82
         9 |    70 |    94
        10 |    79 |    85
        11 |    70 |    95
        12 |    74 |    82
        13 |    76 |   102
        14 |    74 |   116
        15 |    87 |    97
        18 |    92 |   113
        19 |    50 |    95
        20 |    83 |    98
        21 |    73 |   104
        22 |    73 |    86
        23 |    69 |    91
```

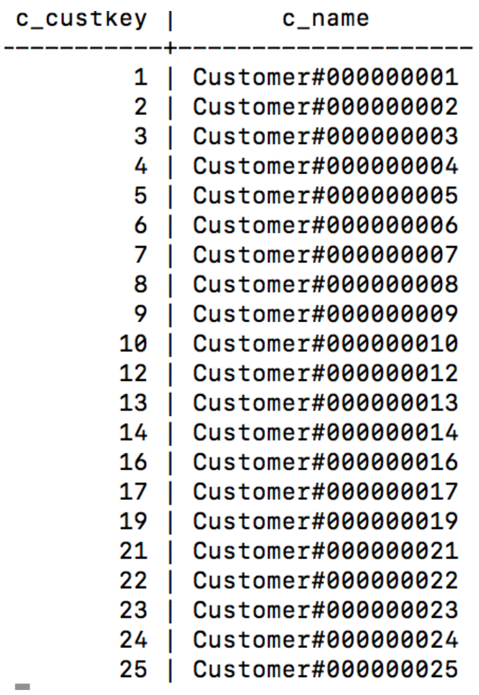
左边的 count 是 1992 年的供应量，右边的 count 是 1993 年的供应量。

(e) 选出来自欧洲地区的 supplier 参与供应的 orders



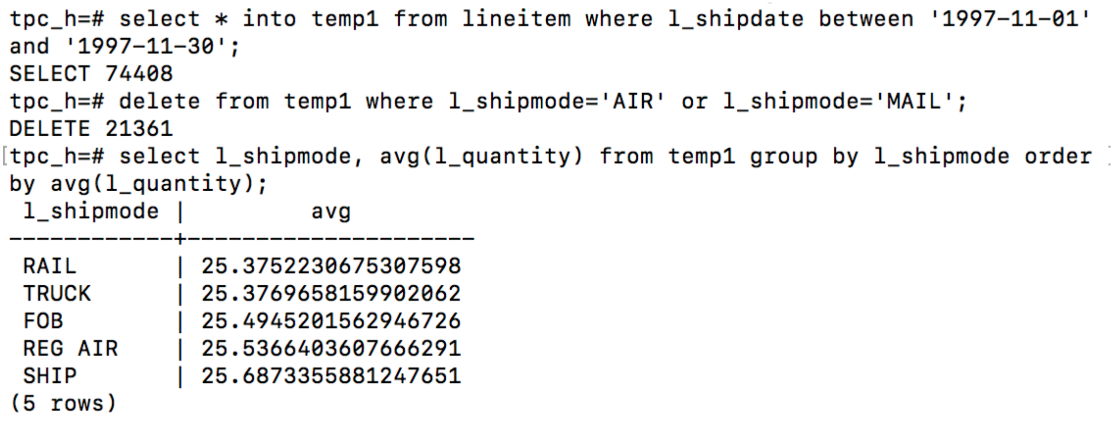
```
tpc_h=# select distinct l_orderkey from lineitem where l_suppkey in (select distinct s_suppkey from supplier where s_nationkey in (select n_nationkey from nation inner join region on r_name='EUROPE' and n_regionkey=r_regionkey));
 l_orderkey
------------
          1
          3
          4
          5
          7
         32
         34
         35
         39
         65
         67
         68
         69
         71
        102
        128
        129
        130
        132
        133
        134
        160
        163
        165
        166
```

(f) 选出不是来自欧洲的 customer



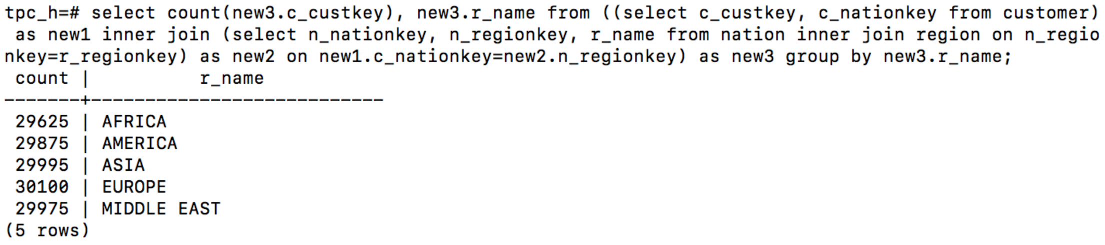
```
 c_custkey |        c_name
-----------+--------------------
         1 | Customer#000000001
         2 | Customer#000000002
         3 | Customer#000000003
         4 | Customer#000000004
         5 | Customer#000000005
         6 | Customer#000000006
         7 | Customer#000000007
         8 | Customer#000000008
         9 | Customer#000000009
        10 | Customer#000000010
        12 | Customer#000000012
        13 | Customer#000000013
        14 | Customer#000000014
        16 | Customer#000000016
        17 | Customer#000000017
        19 | Customer#000000019
        21 | Customer#000000021
        22 | Customer#000000022
        23 | Customer#000000023
        24 | Customer#000000024
        25 | Customer#000000025
```

(g) 选出 1997 年 11 月订购的各类非 ‘AIR’ 和 ‘MAIL’ 型的船的平均质量，并且以生序排列显示。



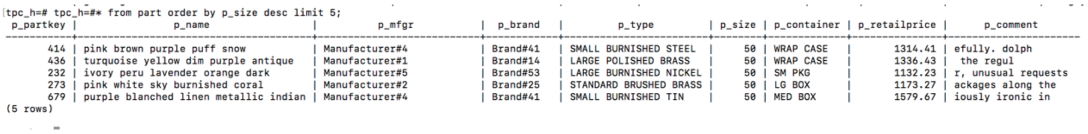
```
tpc_h=# select * into temp1 from lineitem where l_shipdate between '1997-11-01'
and '1997-11-30';
SELECT 74408
tpc_h=# delete from temp1 where l_shipmode='AIR' or l_shipmode='MAIL';
DELETE 21361
tpc_h=# select l_shipmode, avg(l_quantity) from temp1 group by l_shipmode order
by avg(l_quantity);
 l_shipmode |         avg
------------+---------------------
 RAIL       | 25.3752230675307598
 TRUCK      | 25.3769658159902062
 FOB        | 25.4945201562946726
 REG AIR    | 25.5366403607666291
 SHIP       | 25.6873355881247651
(5 rows)
```

(h) 展示每个地区的 customer 的数量。



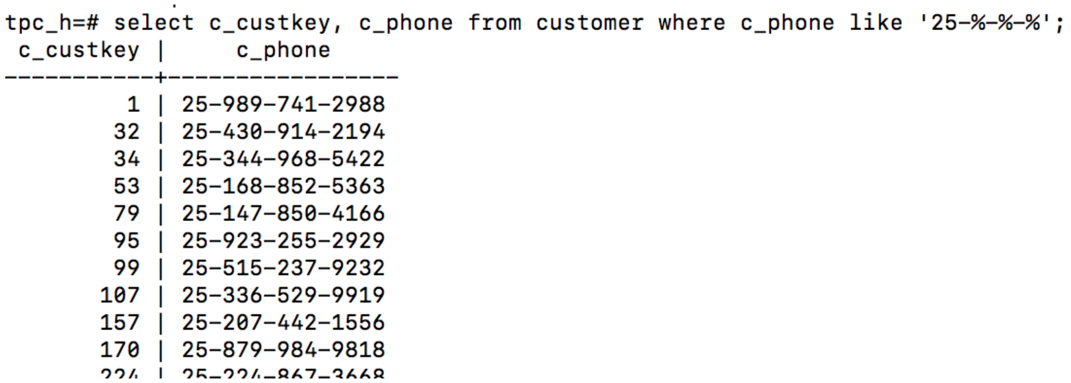
```
tpc_h=# select count(new3.c_custkey), new3.r_name from ((select c_custkey, c_nationkey from customer)
 as new1 inner join (select n_nationkey, n_regionkey, r_name from nation inner join region on n_regio
nkey=r_regionkey) as new2 on new1.c_nationkey=new2.n_regionkey) as new3 group by new3.r_name;
 count |          r_name
-------+---------------------------
 29625 | AFRICA
 29875 | AMERICA
 29995 | ASIA
 30100 | EUROPE
 29975 | MIDDLE EAST
(5 rows)
```

(i) 展示最上端的 5 个 size 最大的 part 的所有信息。



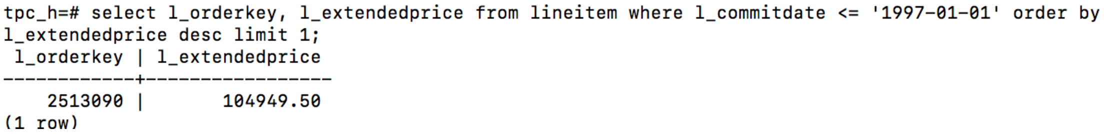
```
tpc_h=# tpc_h=#* from part order by p_size desc limit 5;
 p_partkey |                 p_name                  |           p_mfgr          |  p_brand  |          p_type           | p_size | p_container | p_retailprice |        p_comment
-----------+-----------------------------------------+---------------------------+-----------+---------------------------+--------+-------------+---------------+-------------------------
       414 | pink brown purple puff snow             | Manufacturer#4            | Brand#41  | SMALL BURNISHED STEEL     |     50 | WRAP CASE   |       1314.41 | efully. dolph
       436 | turquoise yellow dim purple antique     | Manufacturer#1            | Brand#14  | LARGE POLISHED BRASS      |     50 | WRAP CASE   |       1336.43 |  the regul
       232 | ivory peru lavender orange dark         | Manufacturer#5            | Brand#53  | LARGE BURNISHED NICKEL    |     50 | SM PKG      |       1132.23 | r, unusual requests
       273 | pink white sky burnished coral          | Manufacturer#2            | Brand#25  | STANDARD BRUSHED BRASS    |     50 | LG BOX      |       1173.27 | ackages along the
       679 | purple blanched linen metallic indian   | Manufacturer#4            | Brand#41  | SMALL BURNISHED TIN       |     50 | MED BOX     |       1579.67 | iously ironic in
(5 rows)
```

(j) 找到电话号码格式为 25-XXX-XXX-XXX 的 customer



```
tpc_h=# select c_custkey, c_phone from customer where c_phone like '25-%-%-%';
 c_custkey |     c_phone
-----------+-----------------
         1 | 25-989-741-2988
        32 | 25-430-914-2194
        34 | 25-344-968-5422
        53 | 25-168-852-5363
        79 | 25-147-850-4166
        95 | 25-923-255-2929
        99 | 25-515-237-9232
       107 | 25-336-529-9919
       157 | 25-207-442-1556
       170 | 25-879-984-9818
       224 | 25-224-867-3668
```

(k) 找到有最大 extend price 的并且在 1997 年之前订购的 part, 并且展示这个最大的 extend price。



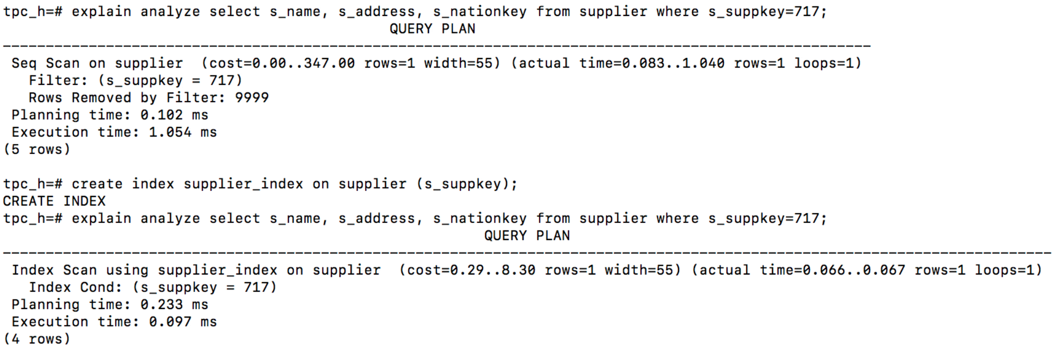
```
tpc_h=# select l_orderkey, l_extendedprice from lineitem where l_commitdate <= '1997-01-01' order by
l_extendedprice desc limit 1;
 l_orderkey | l_extendedprice
------------+-----------------
    2513090 |       104949.50
(1 row)
```

Task3:

首先是默认的 b-tree 索引：

通过建立索引，然后重新执行 a,j,k 题的语句，并分析执行时间。

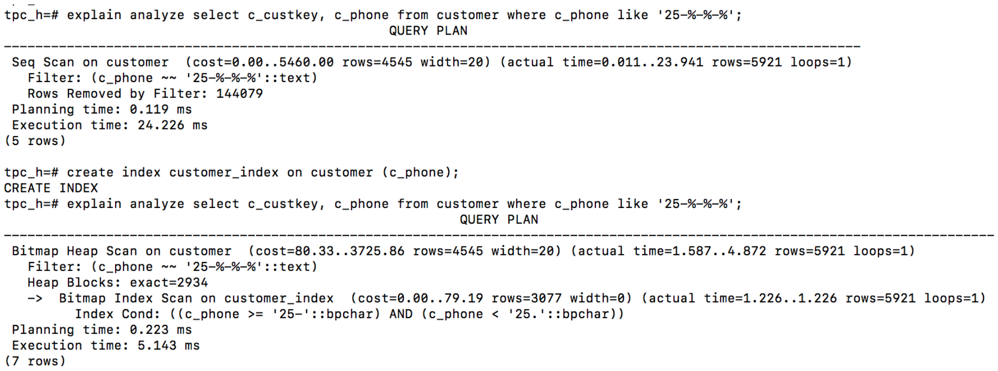
(a)题的结果，建立索引前和建立索引后：



```
tpc_h=# explain analyze select s_name, s_address, s_nationkey from supplier where s_suppkey=717;
                                               QUERY PLAN
-----------------------------------------------------------------------------------------------------
 Seq Scan on supplier  (cost=0.00..347.00 rows=1 width=55) (actual time=0.083..1.040 rows=1 loops=1)
   Filter: (s_suppkey = 717)
   Rows Removed by Filter: 9999
 Planning time: 0.102 ms
 Execution time: 1.054 ms
(5 rows)

tpc_h=# create index supplier_index on supplier (s_suppkey);
CREATE INDEX
tpc_h=# explain analyze select s_name, s_address, s_nationkey from supplier where s_suppkey=717;
                                                          QUERY PLAN
-------------------------------------------------------------------------------------------------------------------------
 Index Scan using supplier_index on supplier  (cost=0.29..8.30 rows=1 width=55) (actual time=0.066..0.067 rows=1 loops=1)
   Index Cond: (s_suppkey = 717)
 Planning time: 0.233 ms
 Execution time: 0.097 ms
(4 rows)
```

可以看到建立前 execution time 为 1.054ms，建立后为 0.097ms，明显缩短了时间。

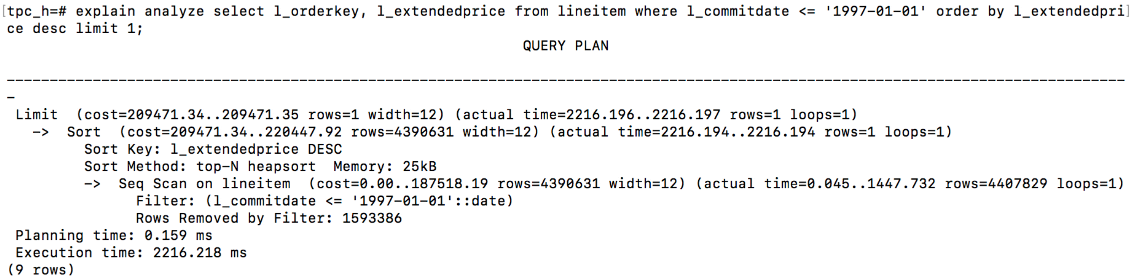
(j)题的结果，建立索引前和建立索引后：



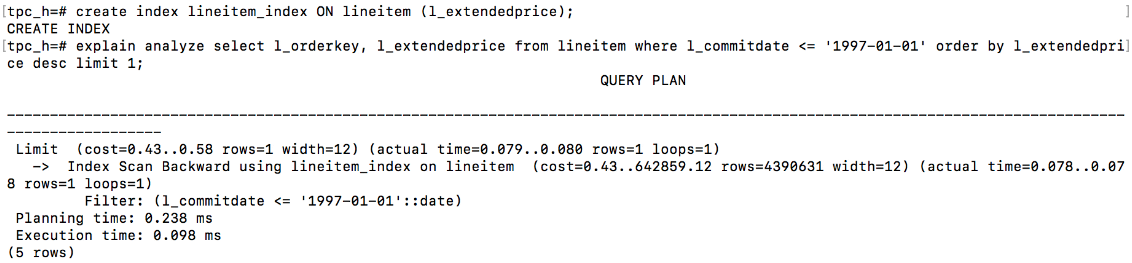
```
tpc_h=# explain analyze select c_custkey, c_phone from customer where c_phone like '25-%-%-%';
                                                  QUERY PLAN
---------------------------------------------------------------------------------------------------------------
 Seq Scan on customer  (cost=0.00..5460.00 rows=4545 width=20) (actual time=0.011..23.941 rows=5921 loops=1)
   Filter: (c_phone ~~ '25-%-%-%'::text)
   Rows Removed by Filter: 144079
 Planning time: 0.119 ms
 Execution time: 24.226 ms
(5 rows)

tpc_h=# create index customer_index on customer (c_phone);
CREATE INDEX
tpc_h=# explain analyze select c_custkey, c_phone from customer where c_phone like '25-%-%-%';
                                                           QUERY PLAN
--------------------------------------------------------------------------------------------------------------------------------
 Bitmap Heap Scan on customer  (cost=80.33..3725.86 rows=4545 width=20) (actual time=1.587..4.872 rows=5921 loops=1)
   Filter: (c_phone ~~ '25-%-%-%'::text)
   Heap Blocks: exact=2934
   ->  Bitmap Index Scan on customer_index  (cost=0.00..79.19 rows=3077 width=0) (actual time=1.226..1.226 rows=5921 loops=1)
         Index Cond: ((c_phone >= '25-'::bpchar) AND (c_phone < '25.'::bpchar))
 Planning time: 0.223 ms
 Execution time: 5.143 ms
(7 rows)
```

可以明显的看出 execution time 的区别，建立索引后的的增速。

(k)题的结果，建立索引前和建立索引后：



```
[tpc_h=# explain analyze select l_orderkey, l_extendedprice from lineitem where l_commitdate <= '1997-01-01' order by l_extendedpri]
ce desc limit 1;
                                                            QUERY PLAN

--------------------------------------------------------------------------------------------------------------------------------------
-
 Limit  (cost=209471.34..209471.35 rows=1 width=12) (actual time=2216.196..2216.197 rows=1 loops=1)
   ->  Sort  (cost=209471.34..220447.92 rows=4390631 width=12) (actual time=2216.194..2216.194 rows=1 loops=1)
         Sort Key: l_extendedprice DESC
         Sort Method: top-N heapsort  Memory: 25kB
         ->  Seq Scan on lineitem  (cost=0.00..187518.19 rows=4390631 width=12) (actual time=0.045..1447.732 rows=4407829 loops=1)
               Filter: (l_commitdate <= '1997-01-01'::date)
               Rows Removed by Filter: 1593386
 Planning time: 0.159 ms
 Execution time: 2216.218 ms
(9 rows)
```

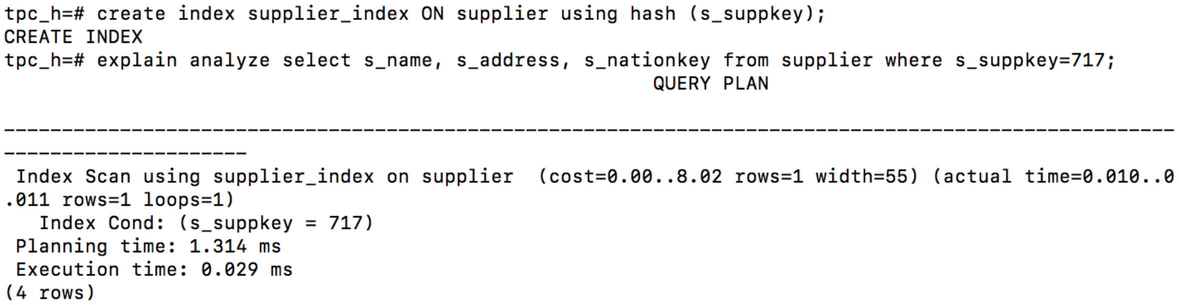


```
[tpc_h=# create index lineitem_index ON lineitem (l_extendedprice);                                                                  ]
CREATE INDEX
[tpc_h=# explain analyze select l_orderkey, l_extendedprice from lineitem where l_commitdate <= '1997-01-01' order by l_extendedpri]
ce desc limit 1;
                                                                     QUERY PLAN

--------------------------------------------------------------------------------------------------------------------------------------
------------------
 Limit  (cost=0.43..0.58 rows=1 width=12) (actual time=0.079..0.080 rows=1 loops=1)
   ->  Index Scan Backward using lineitem_index on lineitem  (cost=0.43..642859.12 rows=4390631 width=12) (actual time=0.078..0.07
8 rows=1 loops=1)
         Filter: (l_commitdate <= '1997-01-01'::date)
 Planning time: 0.238 ms
 Execution time: 0.098 ms
(5 rows)
```

这个结果的增速就非常明显了

而使用 hash 索引：

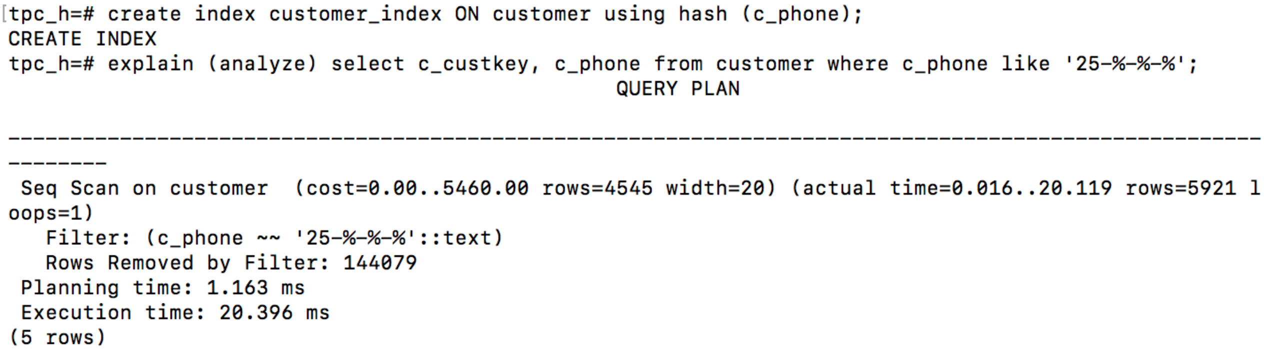
(a)的情况：



```
tpc_h=# create index supplier_index ON supplier using hash (s_suppkey);
CREATE INDEX
tpc_h=# explain analyze select s_name, s_address, s_nationkey from supplier where s_suppkey=717;
                                                   QUERY PLAN

----------------------------------------------------------------------------------------------------
---------------------
 Index Scan using supplier_index on supplier  (cost=0.00..8.02 rows=1 width=55) (actual time=0.010..0
.011 rows=1 loops=1)
   Index Cond: (s_suppkey = 717)
 Planning time: 1.314 ms
 Execution time: 0.029 ms
(4 rows)
```

增速比用 b-tree 的更多。

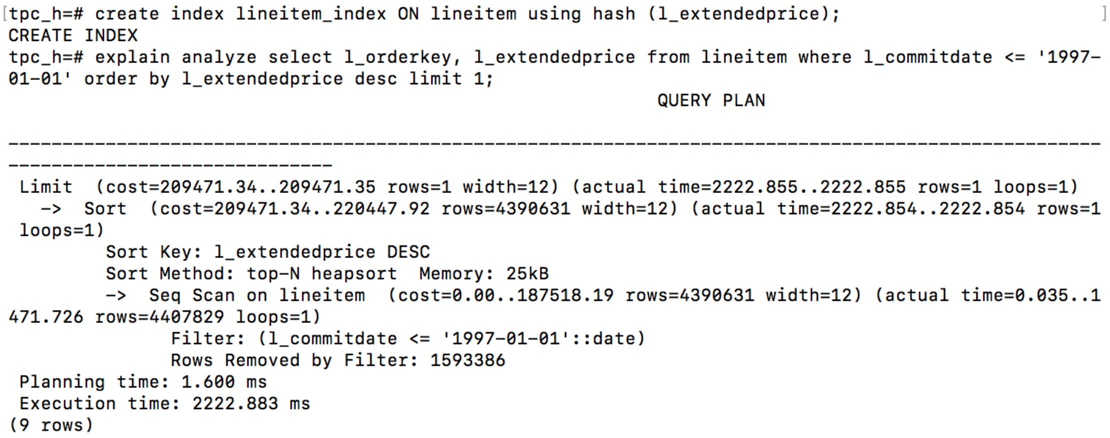
(j)的情况：



```
[tpc_h=# create index customer_index ON customer using hash (c_phone);
CREATE INDEX
tpc_h=# explain (analyze) select c_custkey, c_phone from customer where c_phone like '25-%-%-%';
                                                  QUERY PLAN

----------------------------------------------------------------------------------------------------
--------
 Seq Scan on customer  (cost=0.00..5460.00 rows=4545 width=20) (actual time=0.016..20.119 rows=5921 l
oops=1)
   Filter: (c_phone ~~ '25-%-%-%'::text)
   Rows Removed by Filter: 144079
 Planning time: 1.163 ms
 Execution time: 20.396 ms
(5 rows)
```

不如 b-tree

(k)的情况：



```
tpc_h=# create index lineitem_index ON lineitem using hash (l_extendedprice);
CREATE INDEX
tpc_h=# explain analyze select l_orderkey, l_extendedprice from lineitem where l_commitdate <= '1997-
01-01' order by l_extendedprice desc limit 1;
                                                            QUERY PLAN

----------------------------------------------------------------------------------------------------
------------------------------
 Limit  (cost=209471.34..209471.35 rows=1 width=12) (actual time=2222.855..2222.855 rows=1 loops=1)
   ->  Sort  (cost=209471.34..220447.92 rows=4390631 width=12) (actual time=2222.854..2222.854 rows=1
 loops=1)
         Sort Key: l_extendedprice DESC
         Sort Method: top-N heapsort  Memory: 25kB
         ->  Seq Scan on lineitem  (cost=0.00..187518.19 rows=4390631 width=12) (actual time=0.035..1
471.726 rows=4407829 loops=1)
               Filter: (l_commitdate <= '1997-01-01'::date)
               Rows Removed by Filter: 1593386
 Planning time: 1.600 ms
 Execution time: 2222.883 ms
(9 rows)
```

不如 b-tree

综上可以看出，当需要查找某个具体的数据项时，hash 索引的速度比 b-tree 索引的效率要高；而需要用到排列和大范围匹配的时候，b-tree 的效率会比 hash 索引的高。