实习1

关系数据库创建和查询

理解概念

- 非结构化数据库
  - 一致性 → 最终一致性
  - 互联网上的某些数据，如微博关注数，并非需要实时一致性，只需在特定时间内达到一致性
- OLTP
  - 联机事务处理，如订票、银行转账、选课、成绩登记、微博转发、点赞等
  - 通常涉及部分数据，对应insert、update和delete语句
- OLAP
  - 联机分析处理，如查询商品在不同地区或时间销售额、微博热榜、成绩统计分析等
  - 通常涉及大量数据，对应select语句

关系创建 – 外键约束

- drop表再创建表问题：运行第一次没问题，运行第二次就出错，但是并不是每个表都是如此，station表有这种问题

```
In [3]: %%sql drop table if exists station;
CREATE TABLE station (
    station_id smallint not null primary key,
    station_name text,
    lat real,
    long real,
    dock_count smallint,
    city text,
    installation_date date,
    zip_code text
);
```

```
   1245                     )
   1246         except BaseException as e:

D:\py\python\envs\python27\lib\site-packages\sqlalchemy\engine\default.pyc in do_execute(self, cursor, statem
ent, parameters, context)
    550
    551     def do_execute(self, cursor, statement, parameters, context=None):
--> 552         cursor.execute(statement, parameters)
    553
    554     def do_execute_no_params(self, cursor, statement, context=None):

InternalError: (psycopg2.InternalError) 错误:  无法删除 表 station 因为有其它对象倚赖它
DETAIL:  在表 trip上的约束trip_start_station_id_fkey 倚赖于 表 station
在表 trip上的约束trip_end_station_id_fkey 倚赖于 表 station
HINT:  使用 DROP .. CASCADE 把倚赖对象一并删除.

[SQL: drop table if exists station;]
(Background on this error at: http://sqlalche.me/e/2j85)
```

数据插入 – 主键约束

- 数据重复插入问题

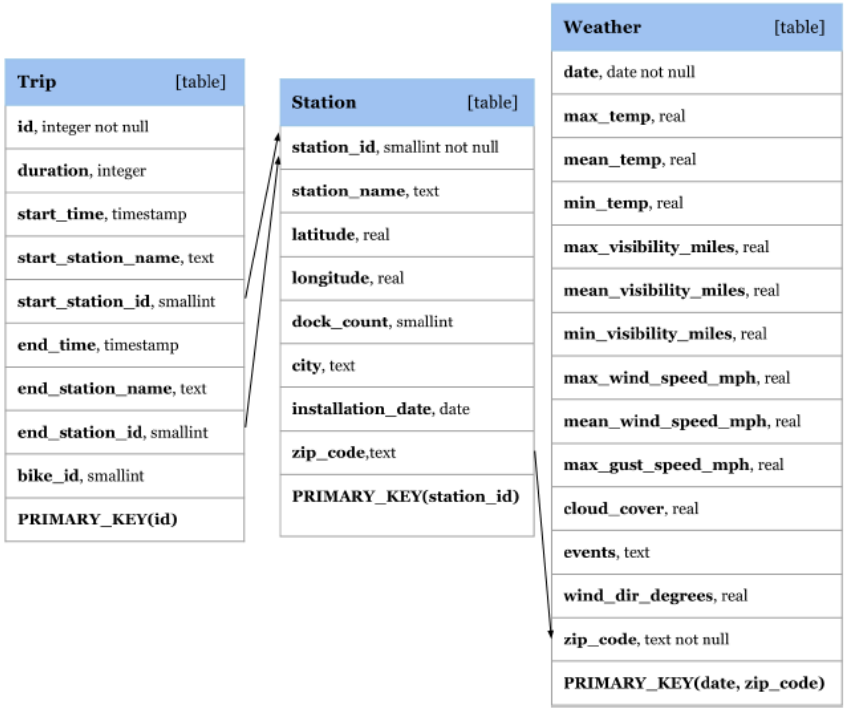
```
In [6]: %sql copy station from 'E://station.txt' delimiter ',';
```

```
-> 1244                         cursor, statement, parameters, context
   1245                     )
   1246         except BaseException as e:

D:\py\python\envs\python27\lib\site-packages\sqlalchemy\engine\default.pyc in do_execute(self, cursor, statem
ent, parameters, context)
    550
    551     def do_execute(self, cursor, statement, parameters, context=None):
--> 552         cursor.execute(statement, parameters)
    553
    554     def do_execute_no_params(self, cursor, statement, context=None):

IntegrityError: (psycopg2.IntegrityError) 错误:  重复键违反唯一约束"station_pkey"
DETAIL:  键值"(station_id)=(2)" 已经存在
CONTEXT:  COPY station, line 1

[SQL: copy station from 'E://station.txt' delimiter ',';]
(Background on this error at: http://sqlalche.me/e/gkpj)
```

- 关系和数据在第一次运行后，就永久保存在数据库中，以后运行时，无需再次创建和插入数据

数据查询与分析

- SQL
  - 课堂练习2(简单/普通模式) → 实习1(普通/困难模式)
  - 看清楚题目要求，题意理解有问题可以在QQ群提问
  - 建议首先使用with简单SQL语句构建临时关系，实现题目要求，再将with语句通过子查询嵌入到select/from/where子句



数据查询与分析

SELECT [ALL|DISTINCT]

<目标列表达式> [别名] [，<目标列表达式> [别名]] …

FROM <表名或视图名> [别名]

[，<表名或视图名> [别名]] …

[WHERE <条件表达式>]

[GROUP BY <列名1>[，<列名2>] …

[HAVING <条件表达式>]]

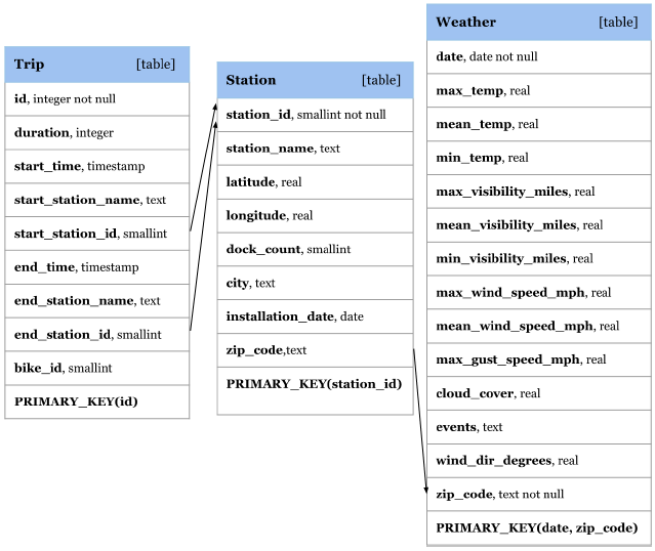
[ORDER BY <列名1> [ASC|DESC]

[，<列名2> [ASC|DESC]] …];

语义上的执行顺序：
1. FROM: 关系笛卡尔积
2. WHERE: 选择满足条件的行
3. GROUP BY: 根据属性分组
4. HAVING: 选择满足条件的组
5. SELECT: 投影需要的列
6. ORDER BY: 结果排序

- $\pi_{A1, A2, \ldots, An}(\sigma_{condition}(R_1 \times R_2 \times \ldots \times R_n))$

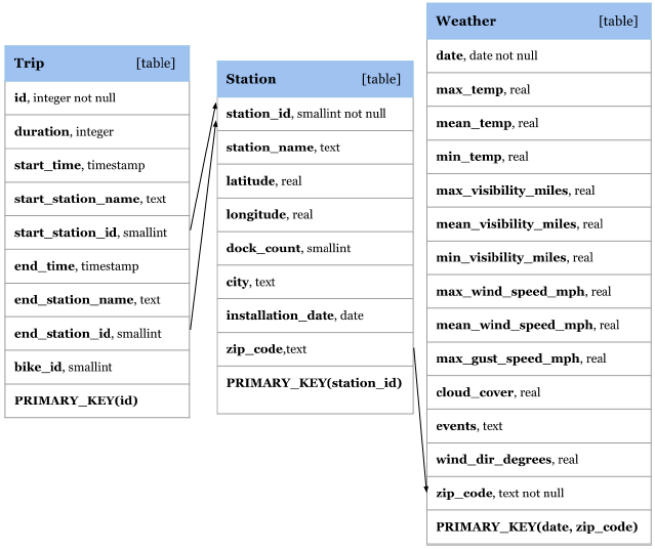
数据查询与分析



- 0. 自行车位最多的站点
  - 最大值问题，SQL常见写法

```
select station_id, dock_count
from station
where dock_count = (select max(dock_count) from station);

select station_id, dock_count
from station
where dock_count >= all(select dock_count from station);

select station_id, dock_count
from station,
        (select max(dock_count) as max_dock_count from station) as mt
where dock_count = max_dock_count
```

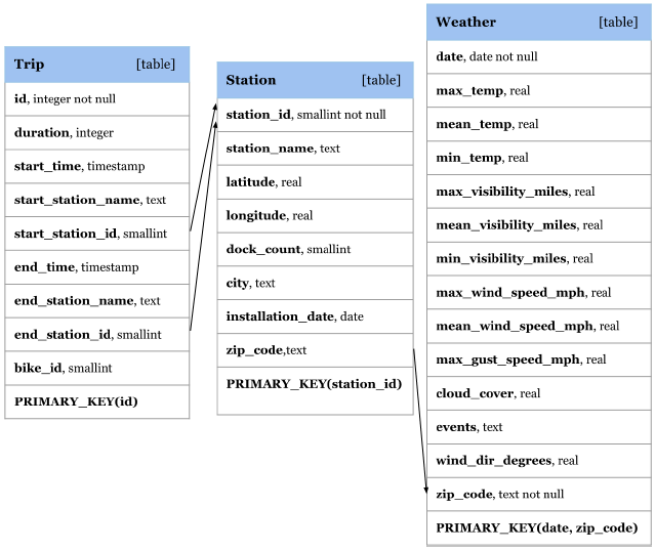
数据查询与分析



- 1. 每个城市站点数
  - 不包含子查询，没有比这更简单了

```
select city, count(station_id) as number
from station
group by city
order by number desc, city asc
```

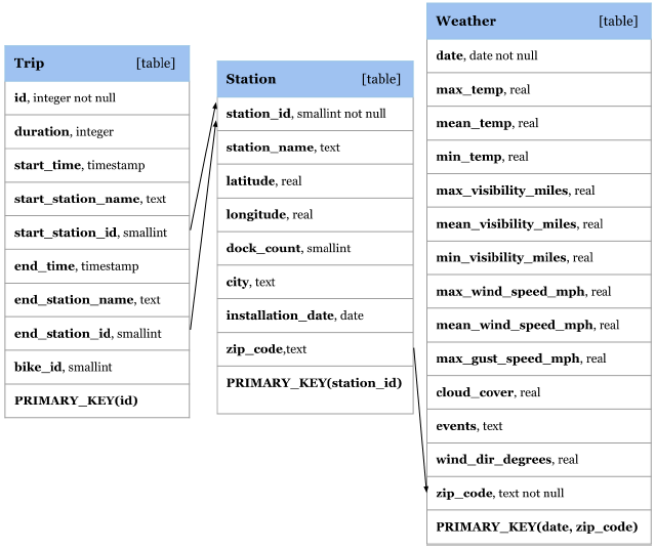
数据查询与分析



- 2. 距离最近的站点对
  - 最小值问题，套用0题模版
  - 不能使用limit，函数使用，去重

```
select S1.station_id as station_id_A,
       S2.station_id as station_id_B,
       dist(S1.lat, S1.long, S2.lat, S2.long) as distance
from station S1, station S2
where S1.station_id < S2.station_id and
       dist(S1.lat, S1.long, S2.lat, S2.long) =
       (select min(dist(S3.lat, S3.long, S4.lat, S4.long))
        from station S3, station S4)
```

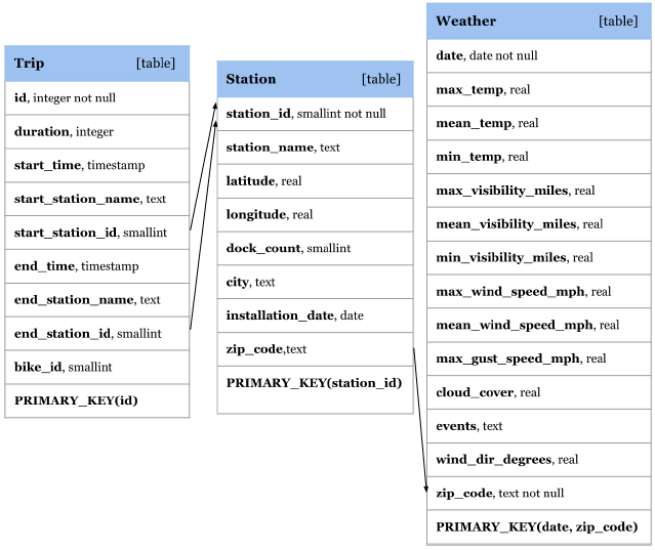
数据查询与分析



- 2. 距离最近的站点对
  - 最小值问题，套用0题模版
  - 不能使用limit，函数使用，去重

```
select S1.station_id as station_id_A,
       S2.station_id as station_id_B,
       dist(S1.lat, S1.long, S2.lat, S2.long) as distance
from station S1, station S2,
     (select min(dist(S3.lat, S3.long, S4.lat, S4.long) as md)
      from station S3, station S4) as D
where S1.station_id < S2.station_id and
      dist(S1.lat, S1.long, S2.lat, S2.long) = md
```

数据查询与分析



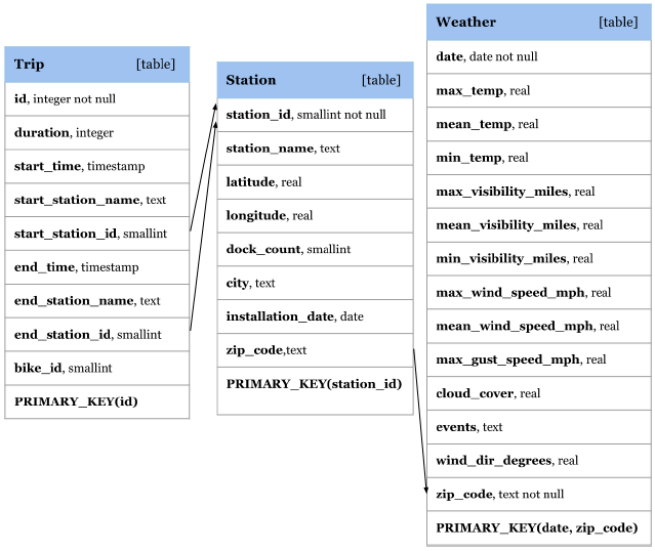
- 2. 每个站点距离最近的站点
  - 每个站点之间的距离

```
select S1.station_id as A,
       S2.station_id as B,
       dist(S1.lat, S1.long, S2.lat, S2.long) as d
from Station S1, Station S2
where S1.station_id <> S2.station_id
```

  - 每个站点选择距离最近的站点

```
select A, B, d
from (…) T1
where d <= any(select d from  (…) as T2 where T1.A = T2.A)
```

数据查询与分析



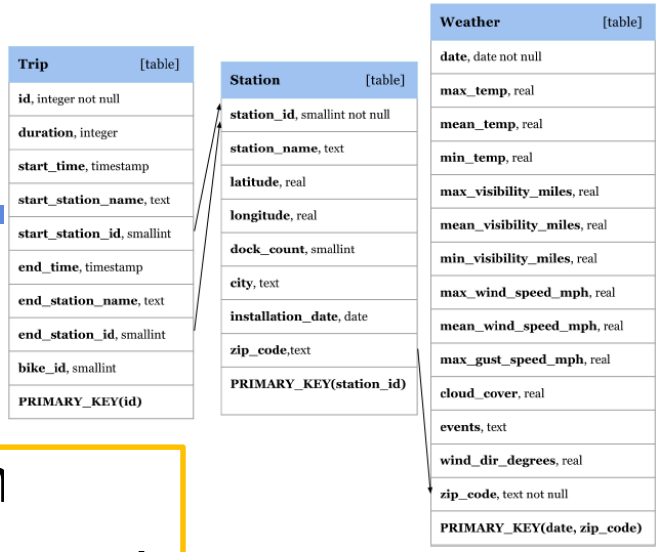
- 2. 每个站点距离最近的站点
  - 每个站点之间的距离 (A, B, d)
  - 每个站点选择距离最近的站点

```
select A, B, d
from (…) T1
where d <= any(select d from  (…) as T2 where T1.A = T2.A)
```

  - 去重(A, B, d)和(B, A, d)

```
select A, B, d
from (…) T1
where d <= any(select d from  (…) as T2 where T2.A = T1.A)
and (A < B or
      d > any(select d from  (…) as T3 where T3.A = T1.B) )
```

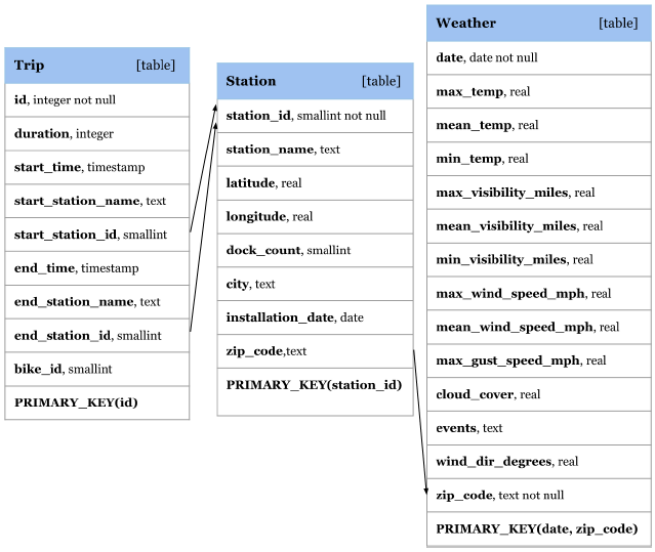
数据查询与分析



- 2. 每个站点距离最近的站点

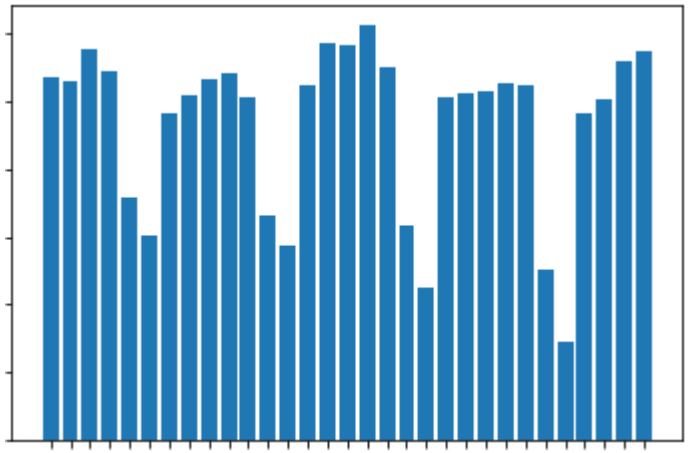
```
select A, B, d
from (select S1.station_id as A, S2.station
      dist(S1.lat, S1.long, S2.lat, S2.long) as d
      from Station S1, Station S2
      where S1.station_id <> S2.station_id) T1
where d <= any(select dist(S3.lat, S3.long, S4.lat, S4.long)
                 from Station S3, Station S4
               where S3.station_id = T1.A and S4.station_id <> T1.A)
and (A < B or
     d > any(select dist(S5.lat, S5.long, S6.lat, S6.long)
               from Station S5, Station S6
              where S5.station_id = T1.B and S6.station_id <> T1.B))
```

数据查询与分析

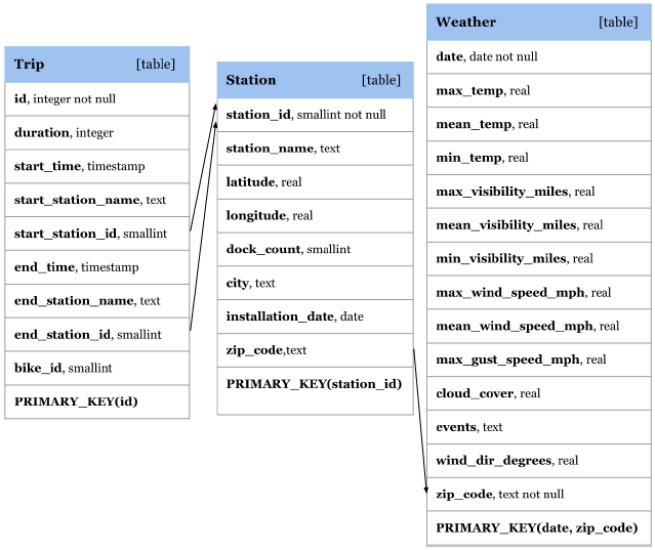


- 3. 2013年10月每天租车量
  - 和1题一样，没有比这更简单了
  - 时间判断和函数使用

```
select date(start_time) as date, count(*) as number
from trip
where start_time > '2013-10-01' and start_time < '2013-11-1'
group by date
order by date
```



数据查询与分析

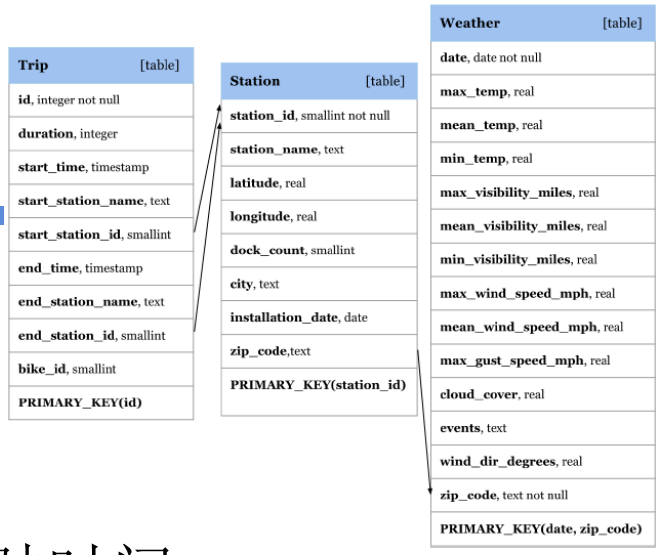


- 4. 租车记录最多的20个站点对
  - 和1题一样，没有比这更简单了

```
select start_station_id, end_station_id, count(*) as trip_count
from trip
group by start_station_id, end_station_id
order by trip_count desc
limit 20
```

  - 查询结果分析
    - 有的对称，有的不对称
    - 原因：从站点A到站点B是下坡，而相反方向是上坡，租车人相对减少；旅游景点站点A-B-C，入口在站点A，出口在站点C；马路两侧都有自行车站点，一侧借车，另一侧还车

数据查询与分析



- 5. 自行车#697的累积行驶时间
  - 自行车#697租车记录

```
select * from trip where bike_id = 697
```

  - 自行车#697还车时间为t时，累积行驶时间

```
select sum(duration) from trip
where bike_id = 697 and end_time <= t
```

  - 自行车#697所有还车时间时，计算累积行驶时间

```
select end_time,
(select sum(duration) from trip T2 where T2.bike_id = 697
and T2.end_time <= T1.end_time) as ctd
from trip T1
where bike_id = 697
order by end_time
```

数据查询与分析

- 选择语句

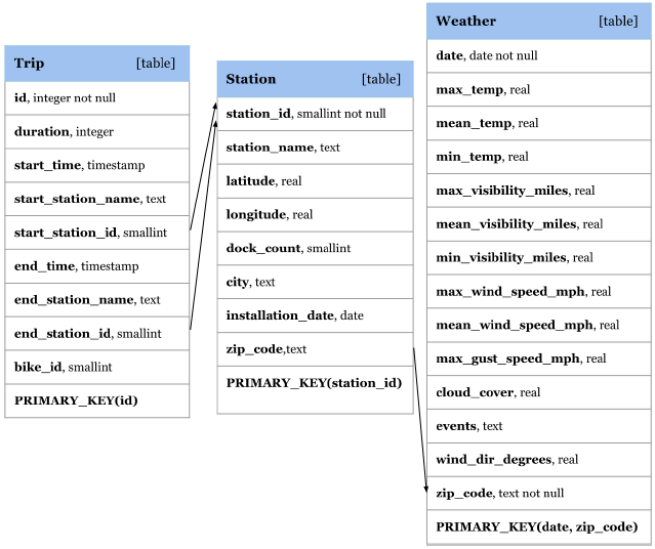
SELECT $A_1, A_2, \ldots, A_n$ #3: what to return

FROM $R_1, R_2, \ldots, R_n$ #1: relations to query

WHERE condition #2: combine, filter relations

```
Answer = {}
for x1 in R1(SQL) do
    for x2 in R2 do
        …..
          for xn in Rn do
              if conditions(x1,…, xn) then (conditions = SQL)
                    A1 = x1.a1
                    A2 = SQL(x1,…, xn) // 要求输出一个值
                    Answer = Answer ∪ {(A1, A2, …, An)}
return Answer
```

数据查询与分析



- 6. 每个城市最受欢迎的站点
  - 分组最大值问题
  - 不区分租车和还车站点

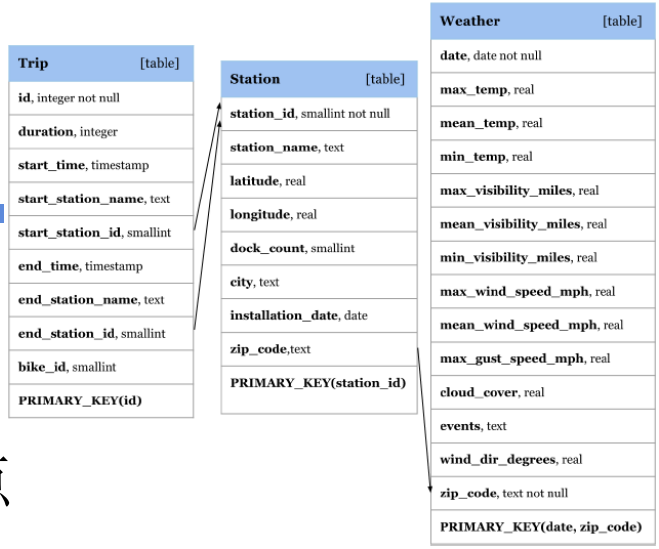
```
with visits as
((select start_station_id as sid from trip)
 union all
 (select end_station_id as sid from trip))
```

思考：可以使用union？

  - 每个站点的使用次数

```
select sid, count(*) from visits group by sid
select city, station_name, count(*)
from station, visits
where station_id = sid
group by city, sid, station_name
```

数据查询与分析



- 6. 每个城市最受欢迎的站点
  - 分组最大值问题
  - 基于城市分组，使用次数最多的站点

```
select city, station_name, count(*)
from station S1, visits
where station_id = sid
group by city, sid, station_name
having count(*) >=
        all(select count(*) from visits
        where sid in (select station_id from station S2
                      where S2.city = S1.city)
        group by sid)
order by city
```

注意：可能存在多种SQL查询方法，嵌套查询 vs. Group By

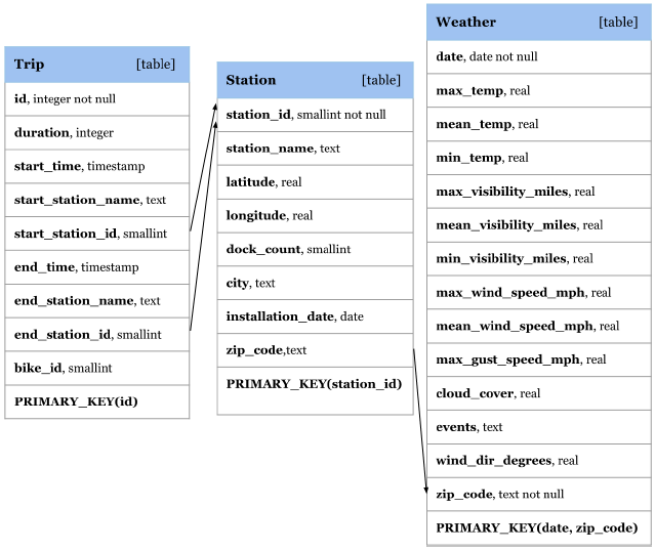
该城市的使用次数最多的站点

数据查询与分析

- 6. 每个城市最受欢迎的站点

city	station_name	count
Mountain View	Mountain View Caltrain Station	13263
Palo Alto	Palo Alto Caltrain Station	3930
Redwood City	Redwood City Caltrain Station	2809
San Francisco	San Francisco Caltrain (Townsend at 4th)	112271
San Jose	San Jose Diridon Caltrain Station	18973

数据查询与分析



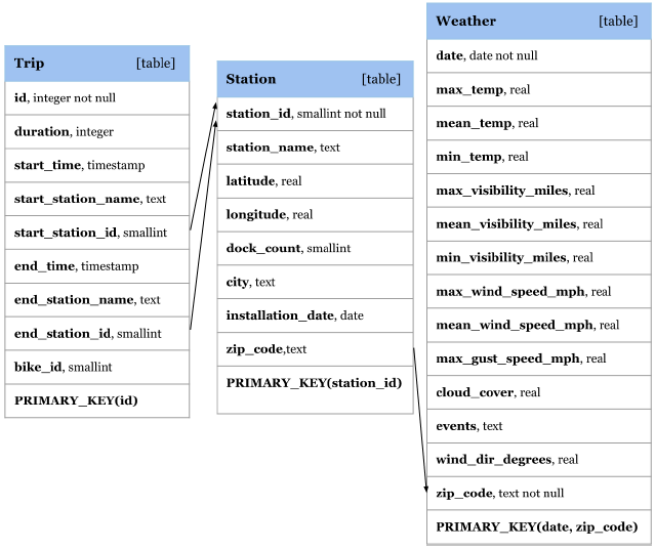
- 7. 每个站点当前的自行车数目
  - Trip只记录了已完成还车的租出记录
  - 假设每辆自行车至少被租出一次，且当前所有车已归还
  - 自行车认为在最后一次还车站点，最后一次还车时间

```
select bike_id, max(end_time) as end_time
from trip
group by bike_id
```

  - 每辆自行车当前在哪个站点

```
select trip.bike_id, end_station_id
from trip, (...) as foo
where trip.bike_id = foo.bike_id and
        trip.end_time = foo.end_time
```

数据查询与分析



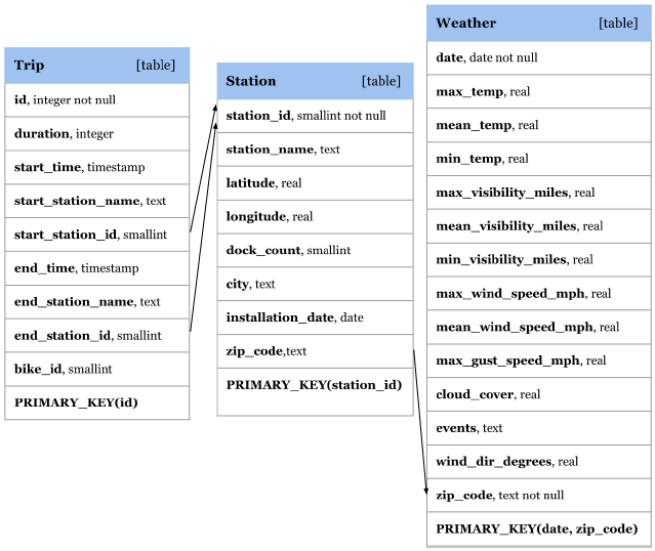
- 7. 每个站点当前的自行车数目
  - 每辆自行车最后一次还车时间
  - 每辆自行车当前在哪个站点

```
select trip.bike_id, end_station_id
from trip, (...) as foo
where trip.bike_id = foo.bike_id and
        trip.end_time = foo.end_time
```

  - 每个站点的自行车数目

```
select end_station_id, count(bike_id) as bike_count
from (...) as temp
group by end_station_id
order by bike_count desc, end_station_id asc
```

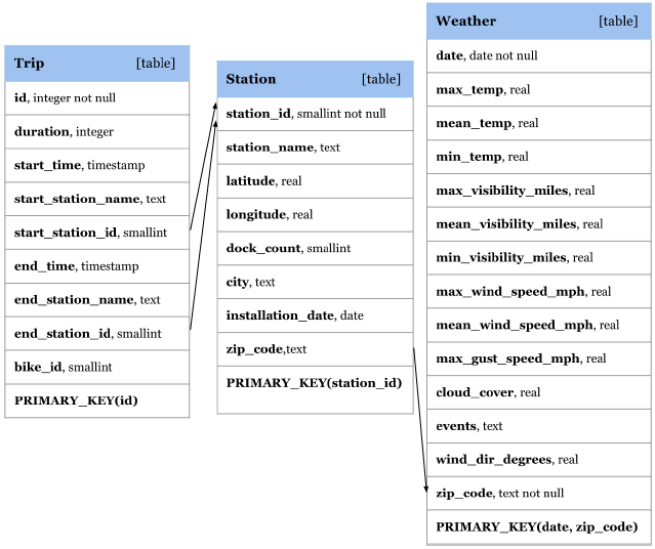
数据查询与分析



- 7. 每个站点当前的自行车数目
  - 每辆自行车最后一次还车时间
  - 每辆自行车当前在哪个站点
  - 每个站点的自行车数目

```
select end_station_id as station_id,
        count(trip.bike_id) as bike_count
from trip,
      (select bike_id, max(end_time) as end_time
       from trip group by bike_id) as foo
where trip.bike_id = foo.bike_id and
       trip.end_time = foo.end_time
group by end_station_id
order by bike_count desc, station_id asc
```

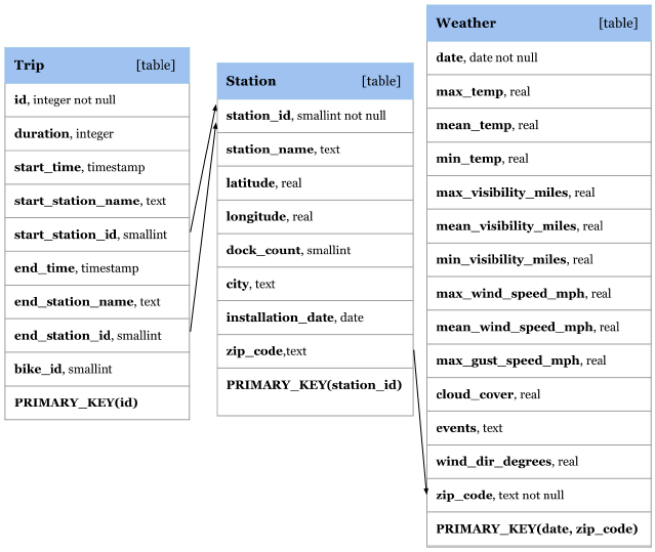
数据查询与分析



- 7. 每个站点当前的自行车数目
  - 每辆自行车最后一次还车时间
  - 每辆自行车当前在哪个站点
  - 每个站点的自行车数目

```
select end_station_id as station_id,
        count(trip.bike_id) as bike_count
from trip natural join
      (select bike_id, max(end_time) as end_time
       from trip group by bike_id) as foo
group by end_station_id
order by bike_count desc, station_id asc
```

数据查询与分析

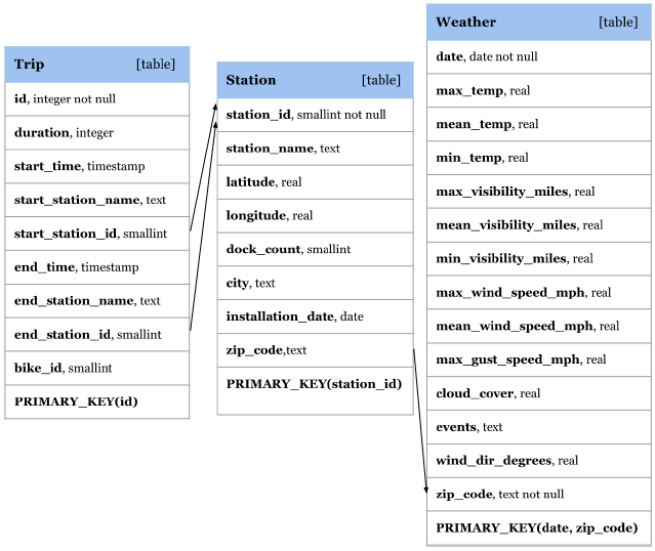


- 7. 每个站点当前的自行车数目
  - 每辆自行车最后一次还车时间
  - 每辆自行车当前在哪个站点
  - 每个站点的自行车数目
  - 当站点车位停满后，无法在该站点还车
  - Trip没有记录人为调度的自行车记录

station_id	dock_count
2	27
61	27
67	27
77	27

station_id	bike_count
69	50
70	45
50	26
55	24
61	23
6	21
2	19
54	18
8	17

数据查询与分析



- 8. 天气与租车关系
  - 不包含子查询，没有比这更简单了
  - 需要关联trip和weather，通过date

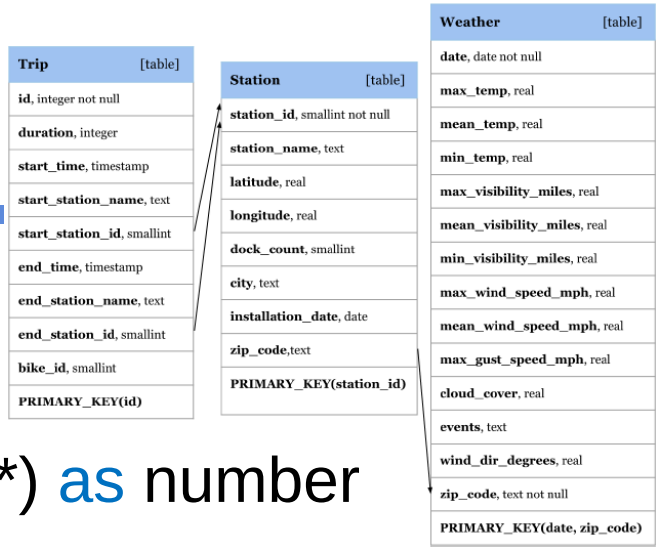
```
select lower(events), count(*) as number
from trip, weather
where date(start_time) = date
group by lower(events)
```

  - 需要关联trip, weather和station，通过station_id, zip_code

```
select lower(events) as events, count(*) as number
from weather, trip, station
where date(start_time) = date and start_station_id =
station_id and station.zip_code = weather.zip_code
group by lower(events)
```

思考：group by用events，结果是否相同？

数据查询与分析



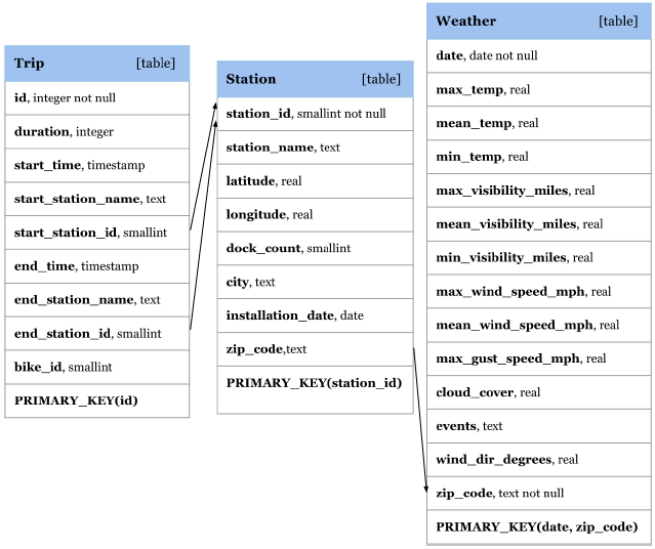
- 8. 天气与租车关系
  - 关联trip, weather和station

```
select lower(events) as events, count(*) as number
from weather, trip, station
where date(start_time) = date and
        start_station_id = station_id and
        station.zip_code = weather.zip_code
group by lower(events)
```

events	number
fog	43676
fog-rain	6877
rain	71613
rain-thunderstorm	1475
None	546318

  - 结论
    - 当天气为rain-thunderstorm时，选择租车的可能性最小？
    - 在rain时选择租车的可能性大于在fog时选择租车的可能性？
  - 不能用绝对数量代替平均数量或概率得出结论
    - 城市犯罪人数越多越不安全？省市GDP越高人民越富裕？...

数据查询与分析



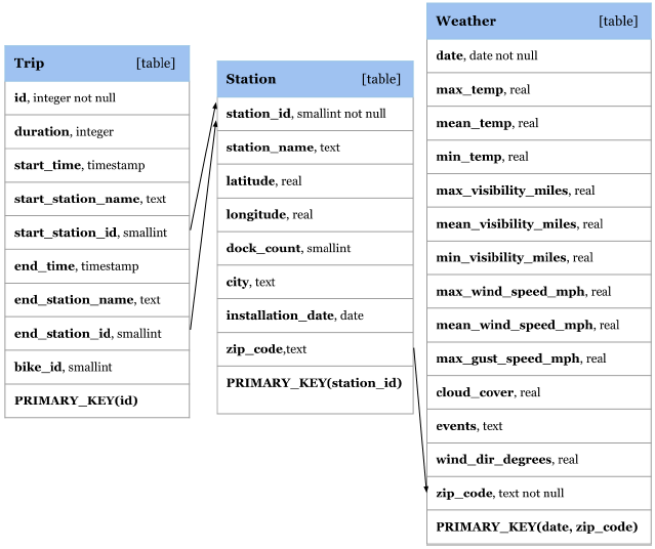
- 8. 天气与租车关系
  - 关联trip, weather和station
  - 不同天气的天数

```
select lower(events) e1, count(*) n1
from weather
group by lower(events)
```

  - 不同天气的租车数量

```
select lower(events) e2, count(*) n2
from weather, trip, station
where date(start_time) = date and
        start_station_id = station_id and
        station.zip_code = weather.zip_code
group by lower(events)
```

数据查询与分析



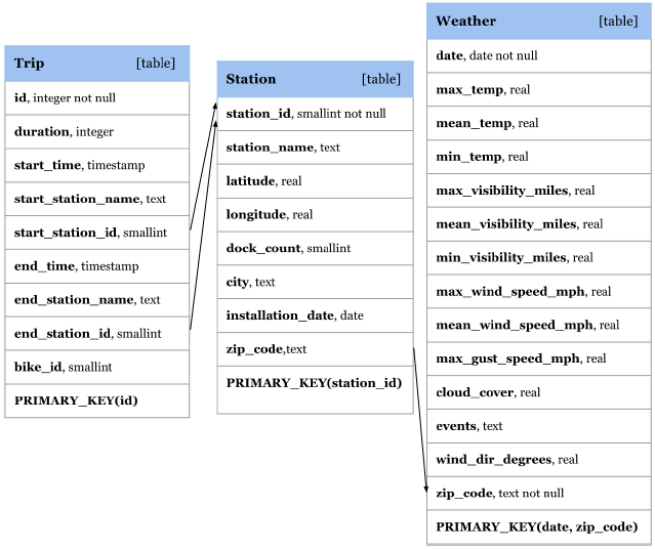
- 8. 天气与租车关系
  - 关联trip, weather和station
  - 不同天气的平均租车数量(self-loop)

```
select e1, round(n2 * 1.0 / n1, 2) from
(select lower(events) e1, count(*) n1
from weather group by lower(events)) as A
(select lower(events) e2, count(*) n2
from weather, trip, station
where date(start_time) = date and start_station_id =
station_id and station.zip_code = weather.zip_
group by lower(events)) as B
where e1 = e2
```

events	number
fog	389.96
fog-rain	404.53
rain	183.62
rain-thunderstorm	491.67

思考：为什么没有None(NULL)的结果？

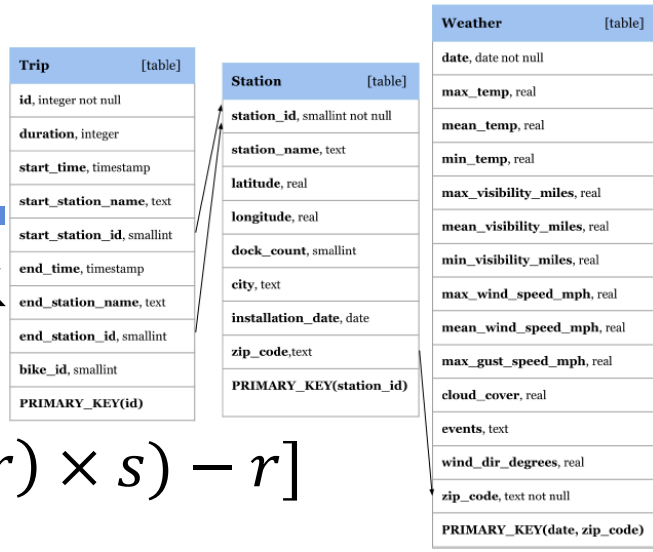
数据查询与分析



- 8. 天气与租车关系
  - 加上NULL之后，有何问题？
  - 如何进一步修改分析？

events	number
fog	389.96
fog-rain	404.53
rain	183.62
rain-thunderstorm	491.67
None	173.82

数据查询与分析



- 9.到达San Jose所有站点的自行车
  - 方法1：除法

$$r \div s = \pi_{(R-S)}(r) - \pi_{(R-S)}[(\pi_{(R-S)}(r) \times s) - r]$$

  - 方法2：数自行车到达的站点数目

```
select count(bike_id)
from station,
(select bike_id, start_station_id as sid from trip
 union
 select bike_id, end_station_id as sid from trip) as temp
where station_id = sid and city = 'San Jose'
group by bike_id
having count(distinct station_id) =
        (select count(*) from station where city = 'San Jose')
```