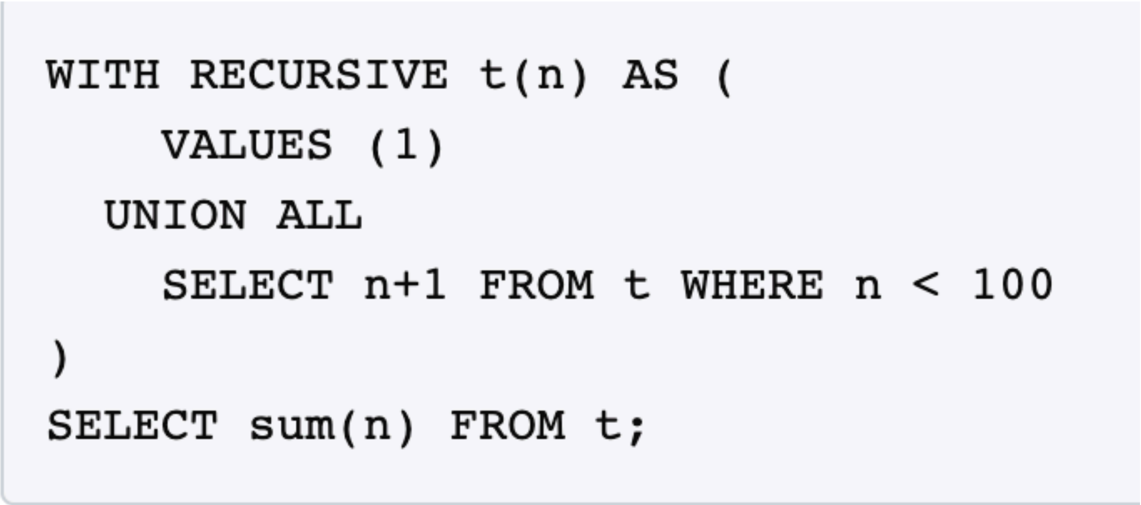
你未必知道的SQL

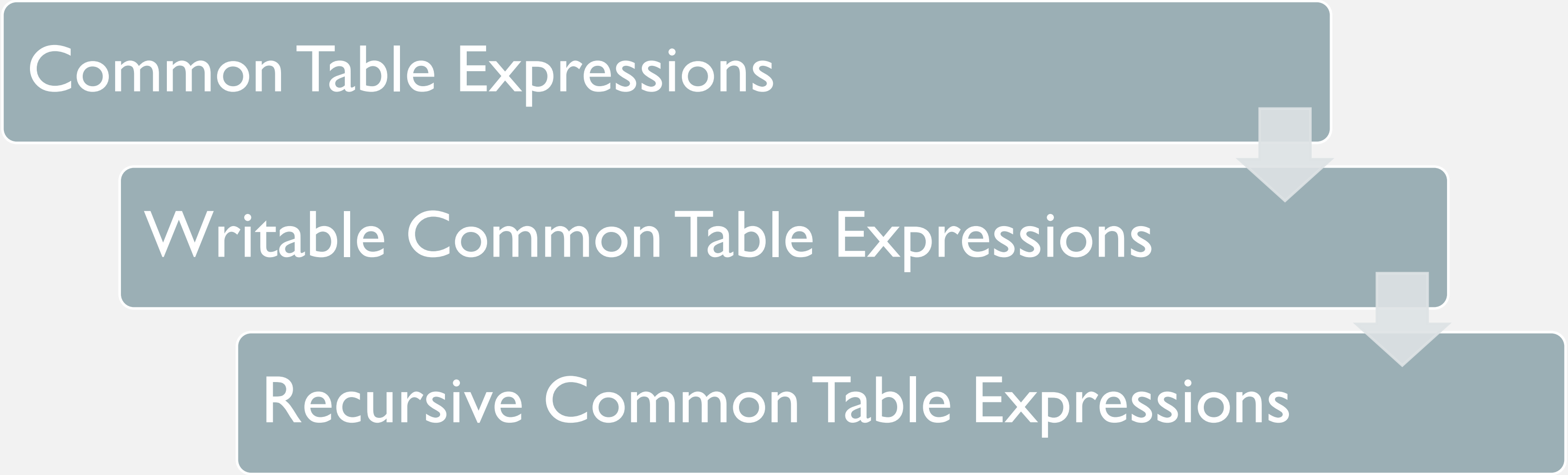
SQL 语言的一些高级技巧



```
WITH RECURSIVE t(n) AS (
    VALUES (1)
  UNION ALL
    SELECT n+1 FROM t WHERE n < 100
)
SELECT sum(n) FROM t;
```

BASIC SAMPLE

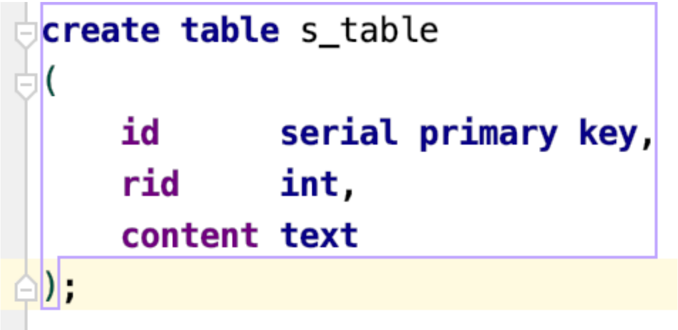
COMMON TABLE EXPRESSIONS



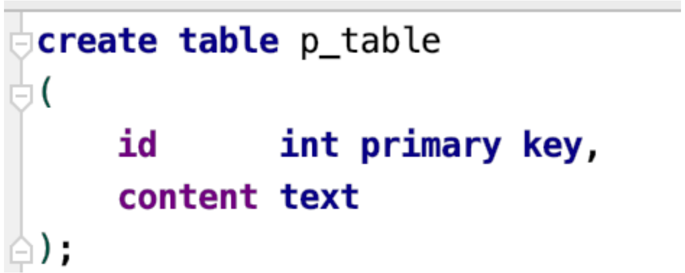
Common Table Expressions

Writable Common Table Expressions

Recursive Common Table Expressions

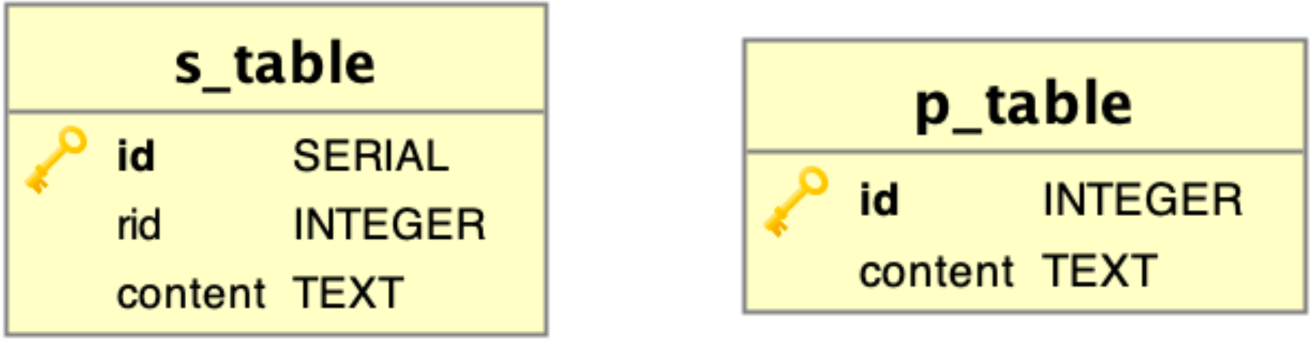


```
create table s_table
(
    id      serial primary key,
    rid     int,
    content text
);
```



```
create table p_table
(
    id      int primary key,
    content text
);
```

ATOMIC ABOUT CTE



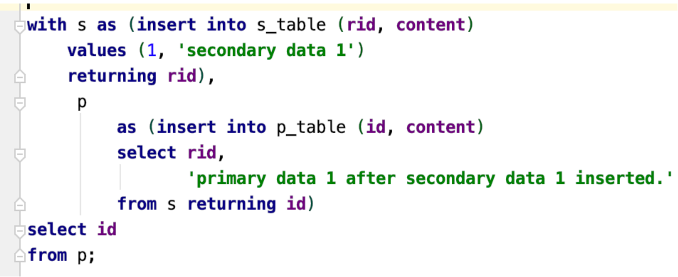
s_table

🔑	**id**	SERIAL
	rid	INTEGER
	content	TEXT

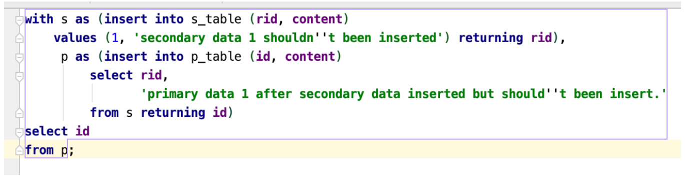
p_table

🔑	**id**	INTEGER
	content	TEXT

ONE PRIMARY & ONE SECONDARY

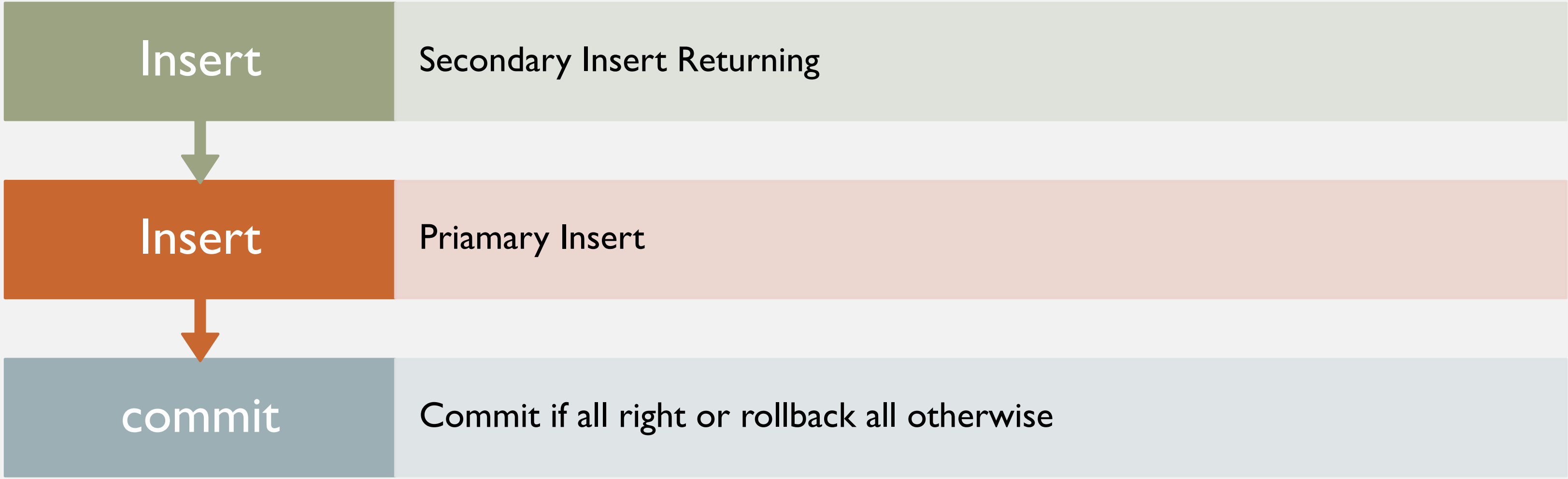


```sql
with s as (insert into s_table (rid, content)
    values (1, 'secondary data 1')
    returning rid),
     p
         as (insert into p_table (id, content)
         select rid,
                'primary data 1 after secondary data 1 inserted.'
         from s returning id)
select id
from p;
```



```sql
with s as (insert into s_table (rid, content)
    values (1, 'secondary data 1 shouldn''t been inserted') returning rid),
     p as (insert into p_table (id, content)
         select rid,
                'primary data 1 after secondary data inserted but should''t been insert.'
         from s returning id)
select id
from p;
```

CTE IS ATOMIC



Insert

Secondary Insert Returning

Insert

Priamary Insert

commit

Commit if all right or rollback all otherwise

CTE IS ATOMIC

业务背景

- 数据按顺序生成
- 异步并发写入
- 仅使用已经“完全写入”的连续空间

实现思路

- Recursive Common Table Expressions
- 获取每个记录的“下一条”

基本思路

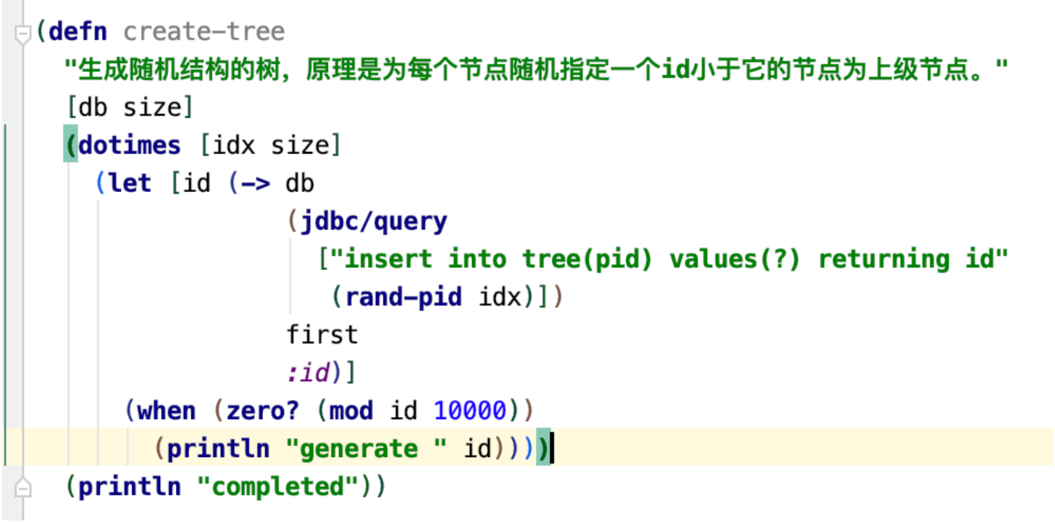
```
with recursive r(id) as (select id
                         from data
                         where id = ?
                         union
                         select d.id
                         from data as d
                                  join r on d.id = r.id + 1)
select data.id, meta, content
from data
         join r on data.id = r.id;
```

更加实用的版本

```
with recursive r(id, meta, content) as (select id, meta,
content
                from data
                where id = ?
                union
                select d.id, d.meta, d.content
                from data as d
                      join r on d.id = r.id + 1)
select id, meta, content
from r limit ?;
```

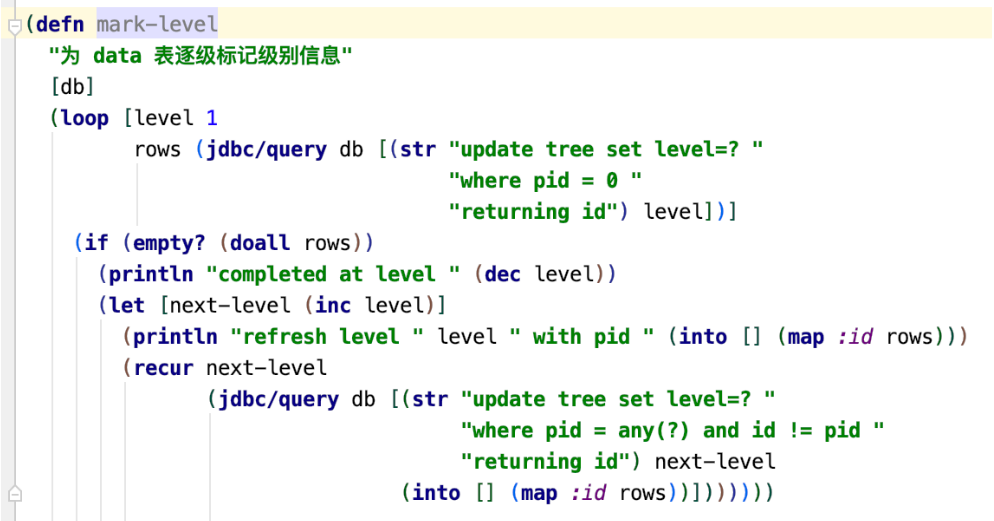
树的经典问题

- 节点回溯
- 树遍历
- 路径染色



```clojure
(defn create-tree
  "生成随机结构的树，原理是为每个节点随机指定一个id小于它的节点为上级节点。"
  [db size]
  (dotimes [idx size]
    (let [id (-> db
                 (jdbc/query
                   ["insert into tree(pid) values(?) returning id"
                    (rand-pid idx)])
                 first
                 :id)]
      (when (zero? (mod id 10000))
        (println "generate " id))))
  (println "completed"))
```

环境准备



```
(defn mark-level
  "为 data 表逐级标记级别信息"
  [db]
  (loop [level 1
         rows (jdbc/query db [(str "update tree set level=? "
                                   "where pid = 0 "
                                   "returning id") level])]
    (if (empty? (doall rows))
      (println "completed at level " (dec level))
      (let [next-level (inc level)]
        (println "refresh level " level " with pid " (into [] (map :id rows)))
        (recur next-level
               (jdbc/query db [(str "update tree set level=? "
                                    "where pid = any(?) and id != pid "
                                    "returning id") next-level
                               (into [] (map :id rows))]))))))
```

环境准备

任意节点回溯——主干

```
with recursive t(id, pid, level) as (select id, pid, level
        from tree
        where level = 29
   union all
   select tree.id, tree.pid, tree.level
   from tree join t on tree.id = t.pid)
select tree.id, tree.pid, meta, tree.level
from tree join t
   on tree.id=t.id;
```

任意节点回溯——最新节点回溯

```
with recursive t(id, pid, level) as (select id, pid, level
        from tree
        where id = 1000000
   union all
   select tree.id, tree.pid, tree.level
   from tree join t on tree.id = t.pid)
select tree.id, tree.pid, meta, tree.level
from tree join t
   on tree.id=t.id;
```

染色(一)

```
with recursive t(id, pid, level) as (select id, pid, level
        from tree
        where id = 1000000
   union all
   select tree.id, tree.pid, tree.level
   from tree join t on tree.id = t.pid)
update tree set meta = '{"path": ["top"]}'::jsonb
from t where tree.id=t.id;
```

染色（二）

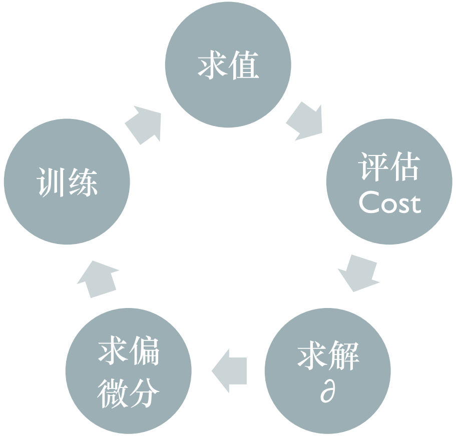
```
with recursive t(id, pid, level) as (select id, pid, level
        from tree
        where id = 1000000
   union all
   select tree.id, tree.pid, tree.level
   from tree join t on tree.id = t.pid)
update tree set meta = jsonb_set(meta, '{path}'::text[],
coalesce(meta #> '{path}', '[]'::jsonb) || '["top"]'::jsonb)
from t where tree.id=t.id;
```

图搜索

梯度下降

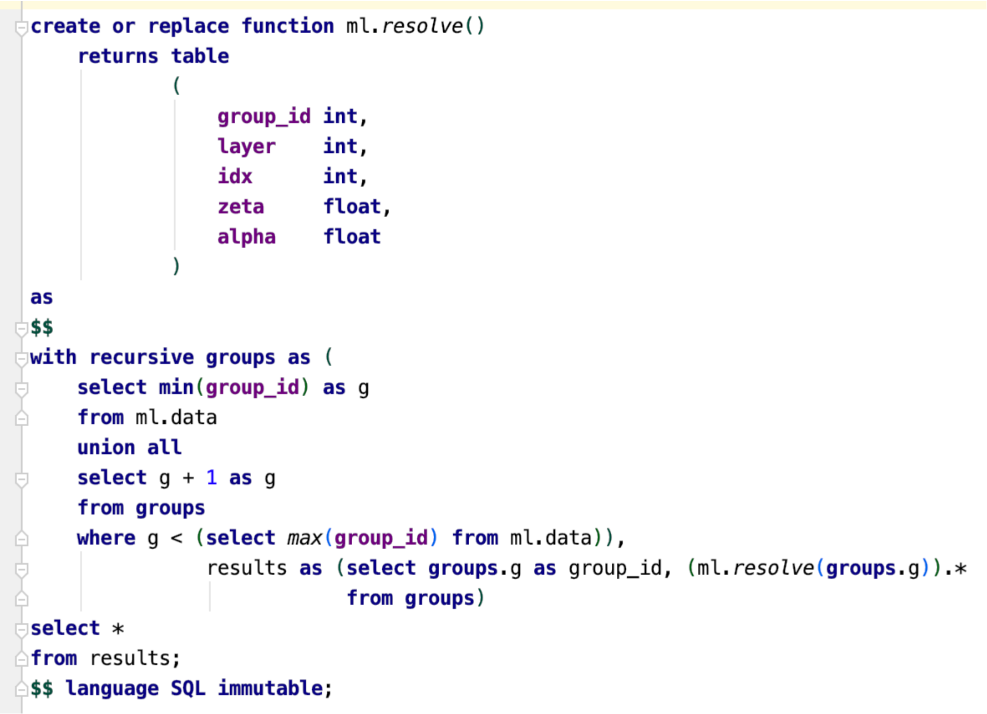
几个外延问题

深度学习——误差反向传播



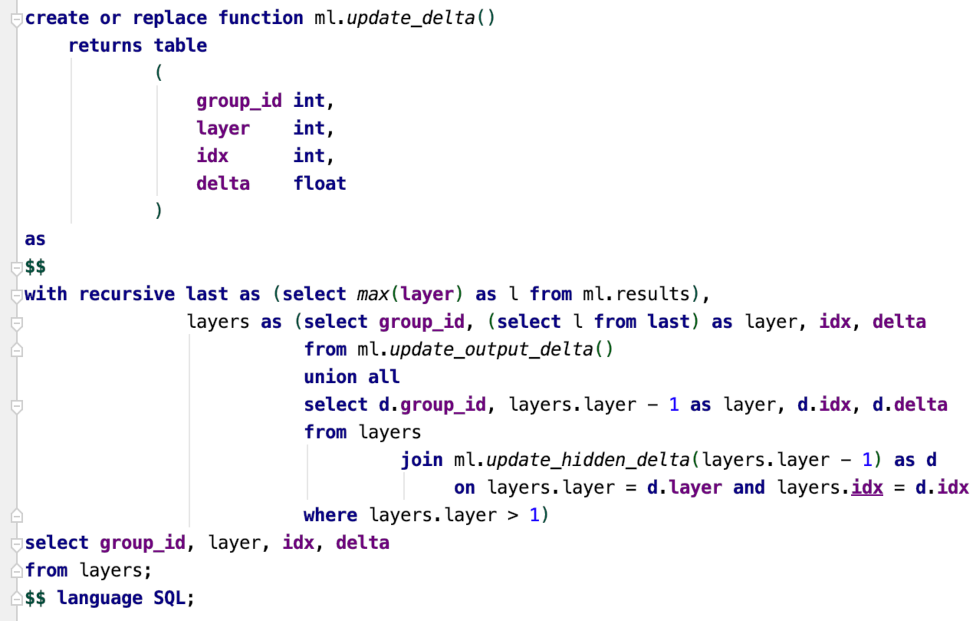
WORKFLOW

RESOLVE



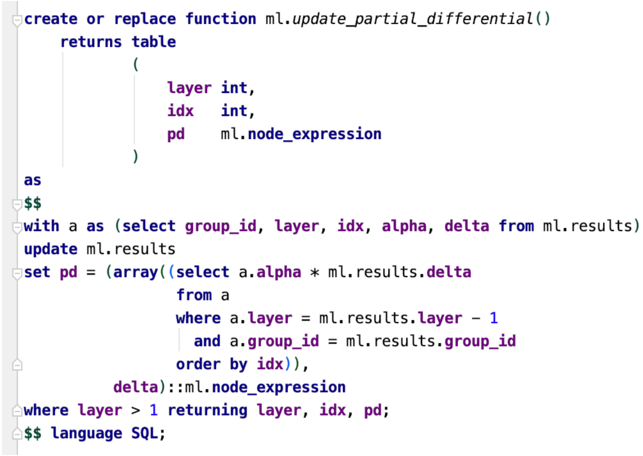
```
create or replace function ml.resolve()
    returns table
            (
                group_id int,
                layer    int,
                idx      int,
                zeta     float,
                alpha    float
            )
as
$$
with recursive groups as (
    select min(group_id) as g
    from ml.data
    union all
    select g + 1 as g
    from groups
    where g < (select max(group_id) from ml.data)),
               results as (select groups.g as group_id, (ml.resolve(groups.g)).*
                           from groups)
select *
from results;
$$ language SQL immutable;
```

∂_n



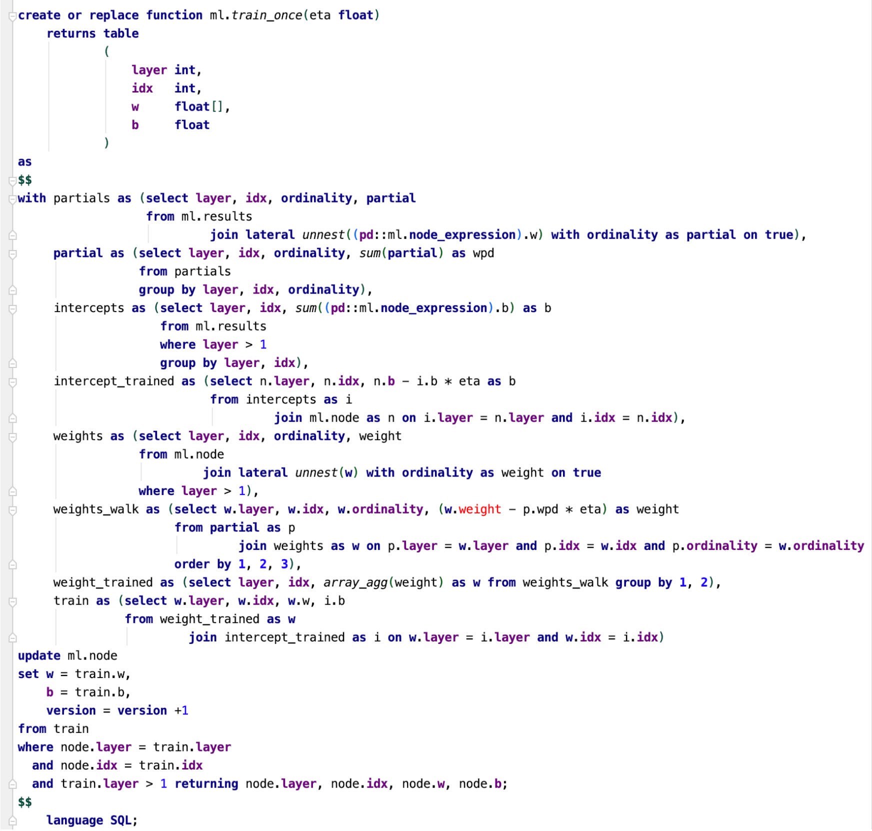
```sql
create or replace function ml.update_delta()
    returns table
            (
                group_id int,
                layer    int,
                idx      int,
                delta    float
            )
as
$$
with recursive last as (select max(layer) as l from ml.results),
               layers as (select group_id, (select l from last) as layer, idx, delta
                          from ml.update_output_delta()
                          union all
                          select d.group_id, layers.layer - 1 as layer, d.idx, d.delta
                          from layers
                                   join ml.update_hidden_delta(layers.layer - 1) as d
                                        on layers.layer = d.layer and layers.idx = d.idx
                          where layers.layer > 1)
select group_id, layer, idx, delta
from layers;
$$ language SQL;
```

$\{\partial C/\partial w_1 ,\ldots,\partial C/\partial w_n \}$ & $\partial C/\partial b$



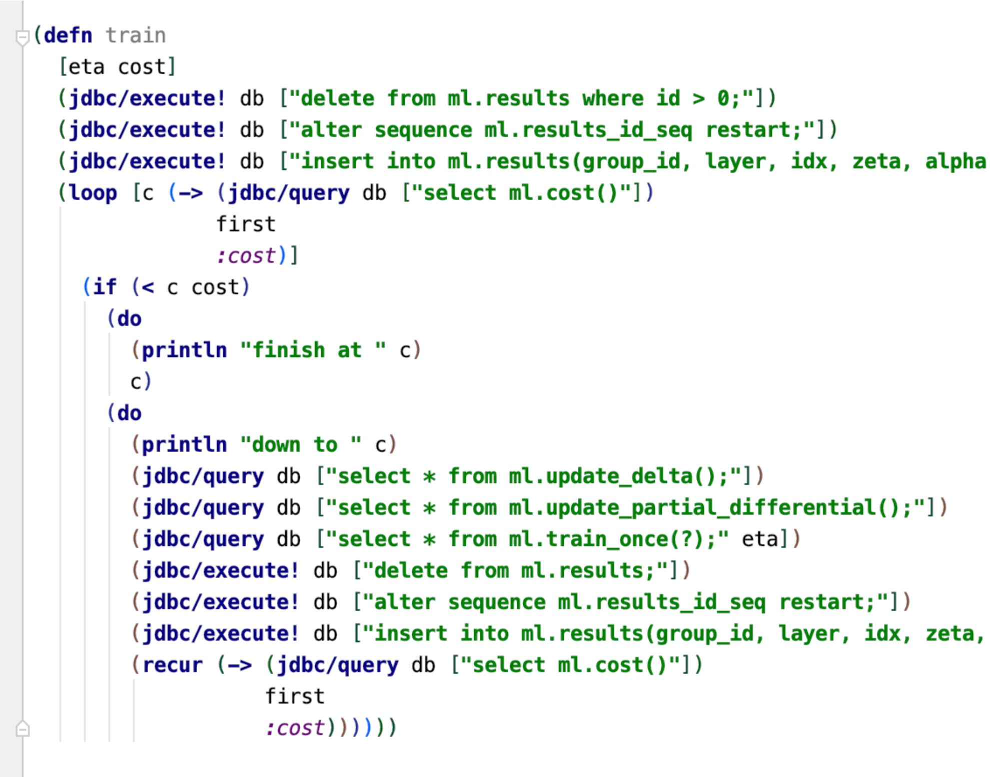
```
create or replace function ml.update_partial_differential()
    returns table
            (
                layer int,
                idx   int,
                pd    ml.node_expression
            )
as
$$
with a as (select group_id, layer, idx, alpha, delta from ml.results)
update ml.results
set pd = (array((select a.alpha * ml.results.delta
                 from a
                 where a.layer = ml.results.layer - 1
                   and a.group_id = ml.results.group_id
                 order by idx)),
          delta)::ml.node_expression
where layer > 1 returning layer, idx, pd;
$$ language SQL;
```

TRAIN ONCE



```
create or replace function ml.train_once(eta float)
    returns table
            (
                layer int,
                idx   int,
                w     float[],
                b     float
            )
as
$$
with partials as (select layer, idx, ordinality, partial
                  from ml.results
                           join lateral unnest((pd::ml.node_expression).w) with ordinality as partial on true),
     partial as (select layer, idx, ordinality, sum(partial) as wpd
                 from partials
                 group by layer, idx, ordinality),
     intercepts as (select layer, idx, sum((pd::ml.node_expression).b) as b
                    from ml.results
                    where layer > 1
                    group by layer, idx),
     intercept_trained as (select n.layer, n.idx, n.b - i.b * eta as b
                           from intercepts as i
                                    join ml.node as n on i.layer = n.layer and i.idx = n.idx),
     weights as (select layer, idx, ordinality, weight
                 from ml.node
                          join lateral unnest(w) with ordinality as weight on true
                 where layer > 1),
     weights_walk as (select w.layer, w.idx, w.ordinality, (w.weight - p.wpd * eta) as weight
                      from partial as p
                               join weights as w on p.layer = w.layer and p.idx = w.idx and p.ordinality = w.ordinality
                      order by 1, 2, 3),
     weight_trained as (select layer, idx, array_agg(weight) as w from weights_walk group by 1, 2),
     train as (select w.layer, w.idx, w.w, i.b
               from weight_trained as w
                        join intercept_trained as i on w.layer = i.layer and w.idx = i.idx)
update ml.node
set w = train.w,
    b = train.b,
    version = version +1
from train
where node.layer = train.layer
  and node.idx = train.idx
  and train.layer > 1 returning node.layer, node.idx, node.w, node.b;
$$
    language SQL;
```

TRAIN



```
(defn train
  [eta cost]
  (jdbc/execute! db ["delete from ml.results where id > 0;"])
  (jdbc/execute! db ["alter sequence ml.results_id_seq restart;"])
  (jdbc/execute! db ["insert into ml.results(group_id, layer, idx, zeta, alpha
  (loop [c (-> (jdbc/query db ["select ml.cost()"])
               first
               :cost)]
    (if (< c cost)
      (do
        (println "finish at " c)
        c)
      (do
        (println "down to " c)
        (jdbc/query db ["select * from ml.update_delta();"])
        (jdbc/query db ["select * from ml.update_partial_differential();"])
        (jdbc/query db ["select * from ml.train_once(?);" eta])
        (jdbc/execute! db ["delete from ml.results;"])
        (jdbc/execute! db ["alter sequence ml.results_id_seq restart;"])
        (jdbc/execute! db ["insert into ml.results(group_id, layer, idx, zeta,
        (recur (-> (jdbc/query db ["select ml.cost()"])
                   first
                   :cost))))))
```

代码

https://github.com/MarchLiu/qcon2019shanghai

Geekbang> 极客邦科技 | InfoQ 极客传媒

InfoQ官网 全新改版上线

促进软件开发领域知识与创新的传播

InfoQ
首页
架构
云计算
AI
运维
前端
QCon十年
软件开发大会
极客时间
登录 | 注册
精选内容
华为面试改革重编程；黑客攻击 Git 私有库索要赎金 | Q 新闻
热点
7 天
1 个月
6 个月
华为面试改革：增加 90 分钟网上编程 30 分钟当面编程环节，充分考察实际动...
快速上手几个 Linux 命令：每家公司都有自己的黑话
京东刘海锋：过去十年架构领域最重要的三个变化
这一天还是来了：亚马逊利用 AI 监控员工工作效率，并生成解雇指令
8 年上市、3 年微服务改造、2 年中台战，回顾这家公司 12 年技术架构转型路
埃森哲被诉：两年花两亿，连个可用的网站都不能交付?
世界第三移动操作系统：KaiOS 的前世今生
技术债务就像俄罗斯方块：我们如何通过修改代码来搞定它?
Facebook：全球最大社交网络，向未知转型
订阅 InfoQ 每周精要
资深编辑为你推荐一周好内容
+订阅
推荐内容
RSS 订阅
MacBook
infoq.cn
Apache Spark 实现可扩展日志分析，挖... 大潜力

关注InfoQ网站
第一时间浏览原创IT新闻资讯

免费下载迷你书
阅读一线开发者的技术干货

问答时间