Insights of Approximate Query Processing Systems

Presented by: Huanyi Chen

Ruoxi Zhang

UNIVERSITY OF WATERLOO

Agenda

- Introduction
- Background
- VerdictDB & SnappyData
- Experiment Setup
- Evaluation
- Insights

Why AQP?

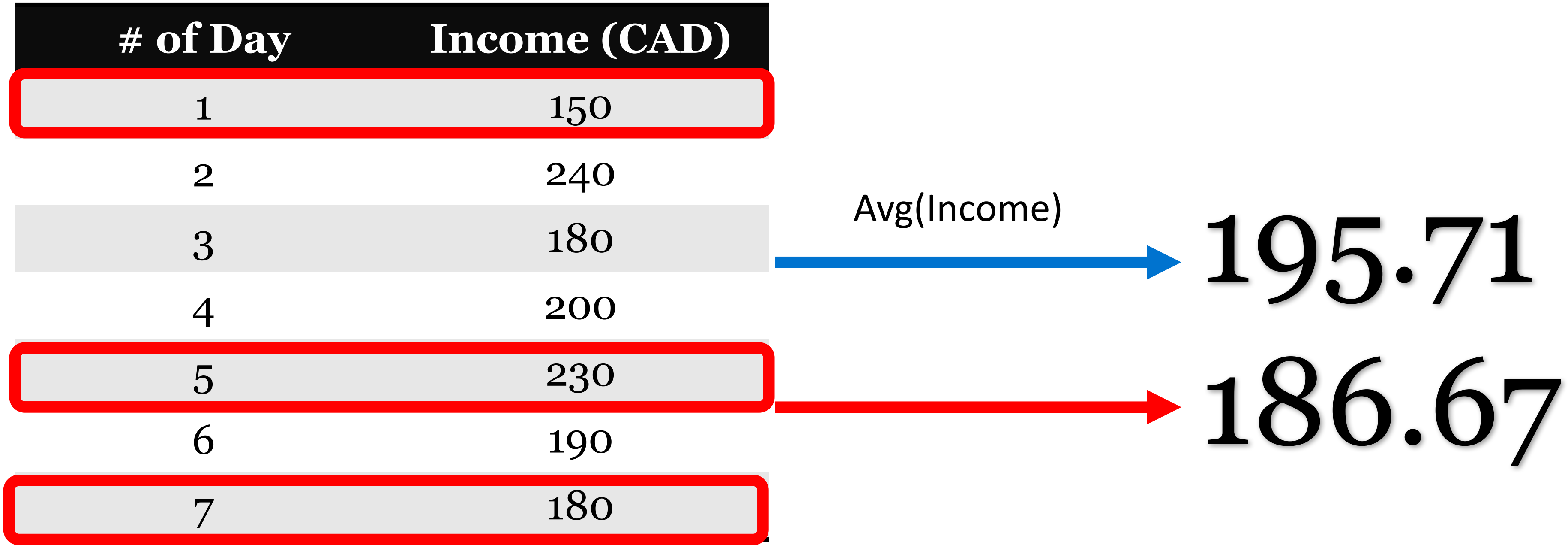
# of Day	Income (CAD)
1	150
2	240
3	180
4	200
5	230
6	190
7	180

shop income

Avg(Income) → 195.71

Why AQP?

more efficient (50% rows)
accuracy > 95%

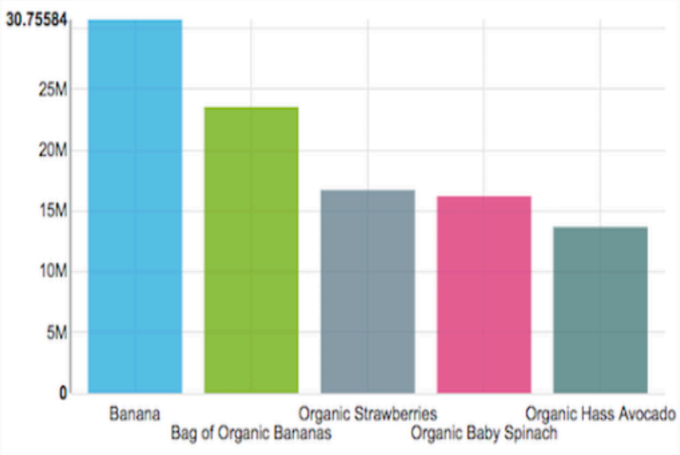


# of Day	Income (CAD)
1	150
2	240
3	180
4	200
5	230
6	190
7	180

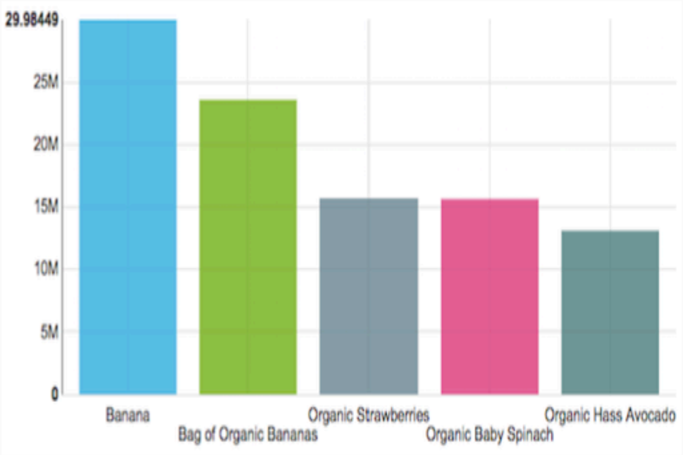
shop income

Why AQP?

99.9% Identical
100x-200x Faster

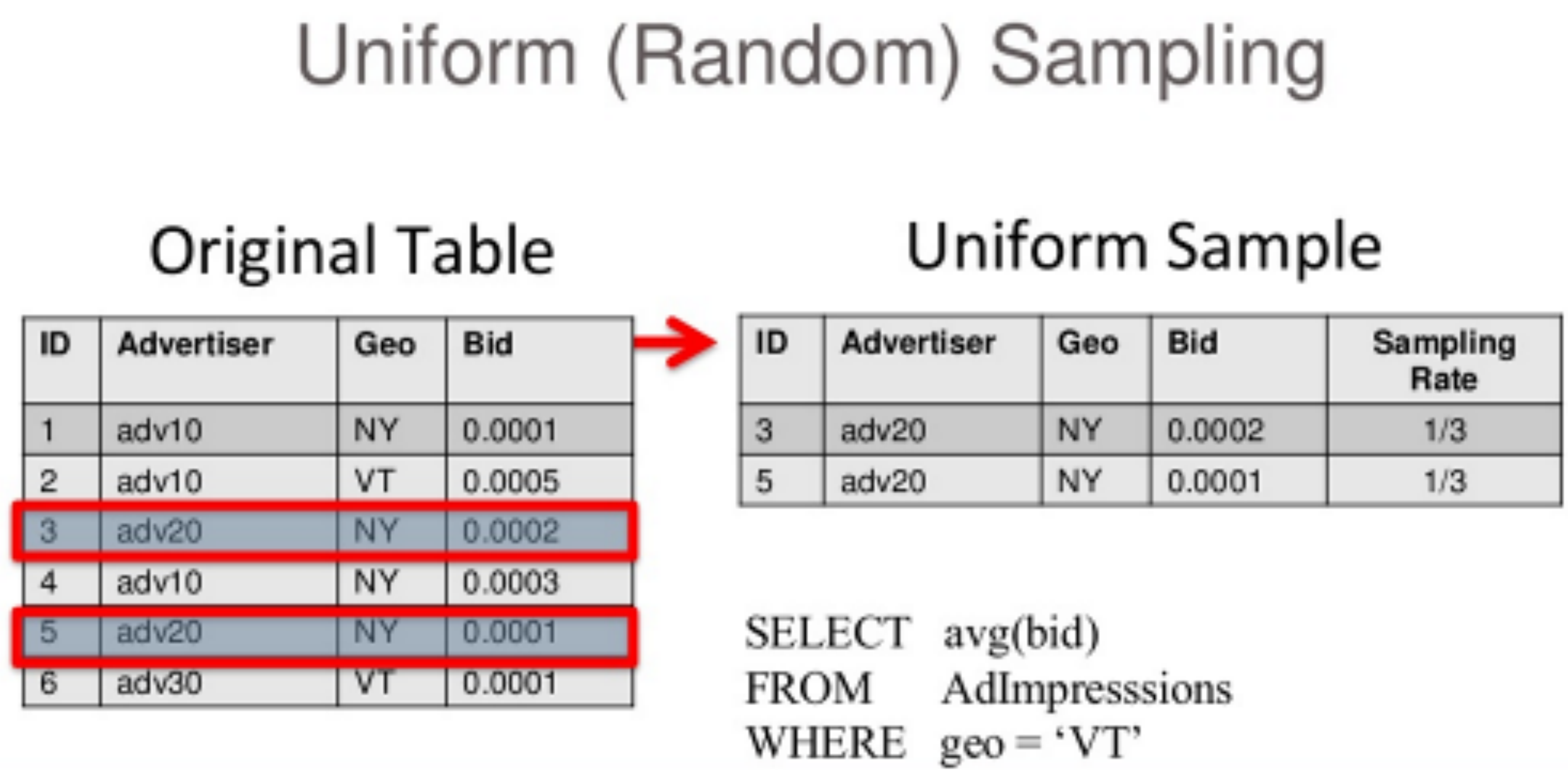


Answer by **Impala** in 350 seconds



Answer by **VerdictDB** in 2 seconds

Sampling Based AQP



Uniform (Random) Sampling

Original Table

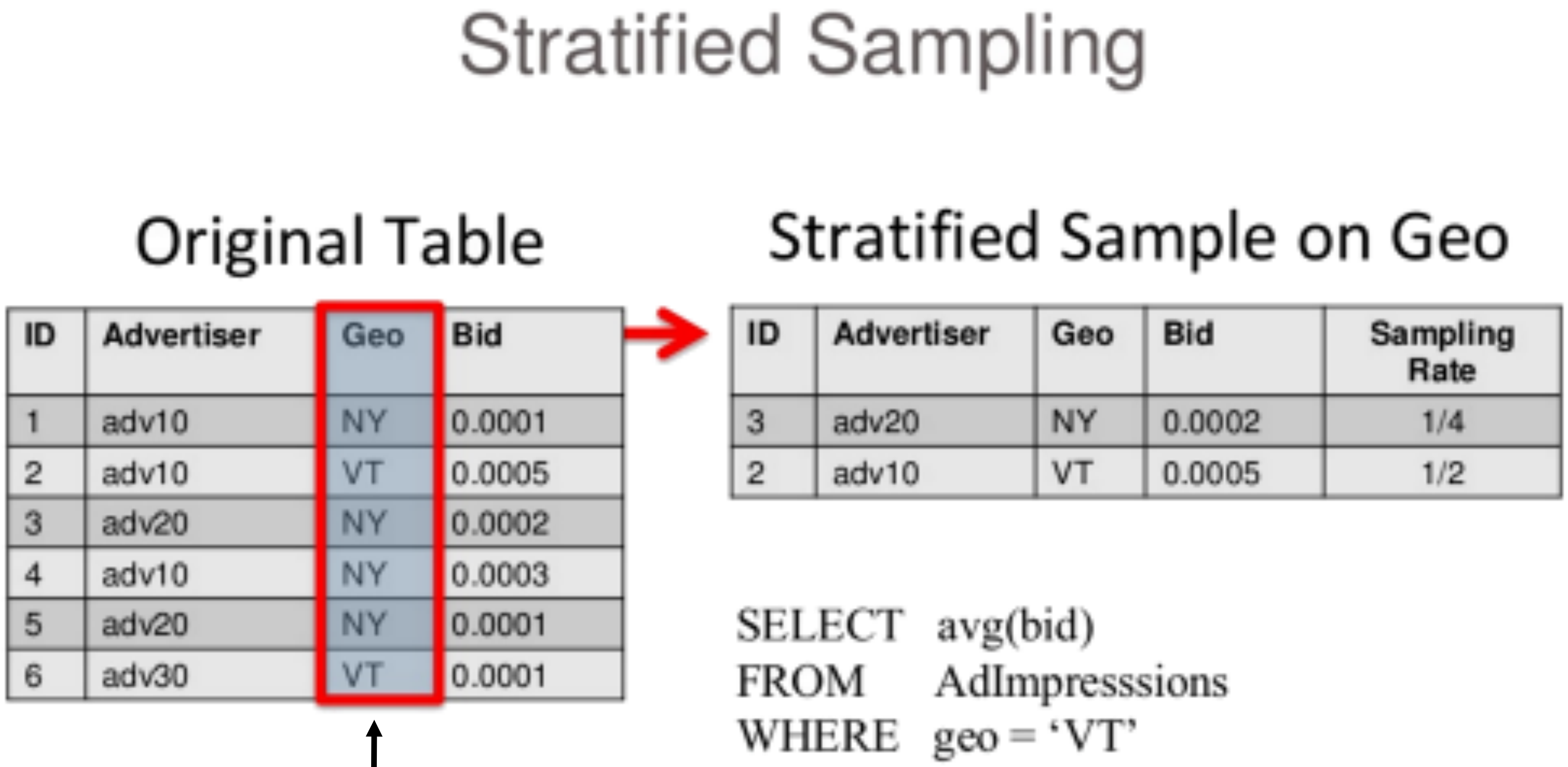
ID	Advertiser	Geo	Bid
1	adv10	NY	0.0001
2	adv10	VT	0.0005
3	adv20	NY	0.0002
4	adv10	NY	0.0003
5	adv20	NY	0.0001
6	adv30	VT	0.0001

Uniform Sample

ID	Advertiser	Geo	Bid	Sampling Rate
3	adv20	NY	0.0002	1/3
5	adv20	NY	0.0001	1/3

```
SELECT   avg(bid)
FROM     AdImpresssions
WHERE    geo = ‘VT’
```

Sampling Based AQP



Stratified Sampling

Original Table

ID	Advertiser	Geo	Bid
1	adv10	NY	0.0001
2	adv10	VT	0.0005
3	adv20	NY	0.0002
4	adv10	NY	0.0003
5	adv20	NY	0.0001
6	adv30	VT	0.0001

Stratified Sample on Geo

ID	Advertiser	Geo	Bid	Sampling Rate
3	adv20	NY	0.0002	1/4
2	adv10	VT	0.0005	1/2

```
SELECT   avg(bid)
FROM     AdImpresssions
WHERE    geo = 'VT'
```

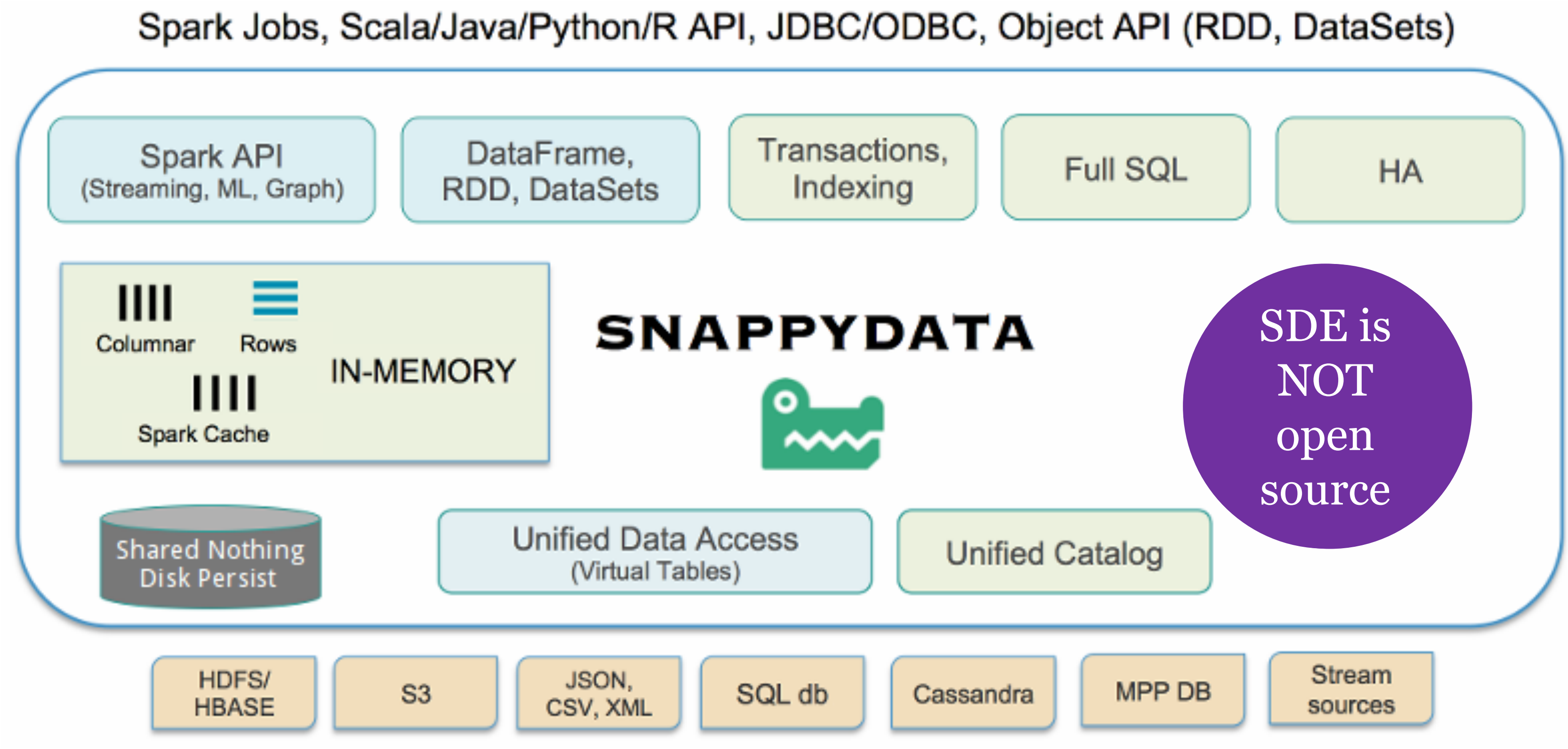
Query Column Set (QCS)

Why SnappyData & VerdictDB ?

- **Spark**
- **Open-source***

Name	Online/ Offline	Distributed/ Standalone	Platform	Algorithm	Skewed
BlinkDB	Offline	Distributed	Hive/Hadoop (Shark)	Stratified sampling	Yes
Sapprox	Online	Distributed	Hadoop	Distribution-aware Online sampling	No
Approxhadoop	Online	Distributed	Hadoop	Approximation-enabled MapReduce	No
Quickr	Online	Distributed	N/A	ASALQA algorithm	No
SnappyData	**Online**	**Distributed**	**Spark and GemFire**	**Spark as a computational engine; GemFire as transactional store**	**No**
FluoDB	Online	Distributed	Spark	Mini-batch execution OLA Model	No
XDB	Online	Standalone	PostgreSQL	Wander join	No
VerdictDB	**Online**	**Standalone**	**Spark SQL**	**Database learning**	**No**
IDEA	Online	Standalone	N/A	Reuse answers of past overlapping queries for new query	No
BEAS	Online	Standalone	Commercial DBMS	Approximability theorem	No
ABS	Online	Standalone	N/A	Bootstrap	No

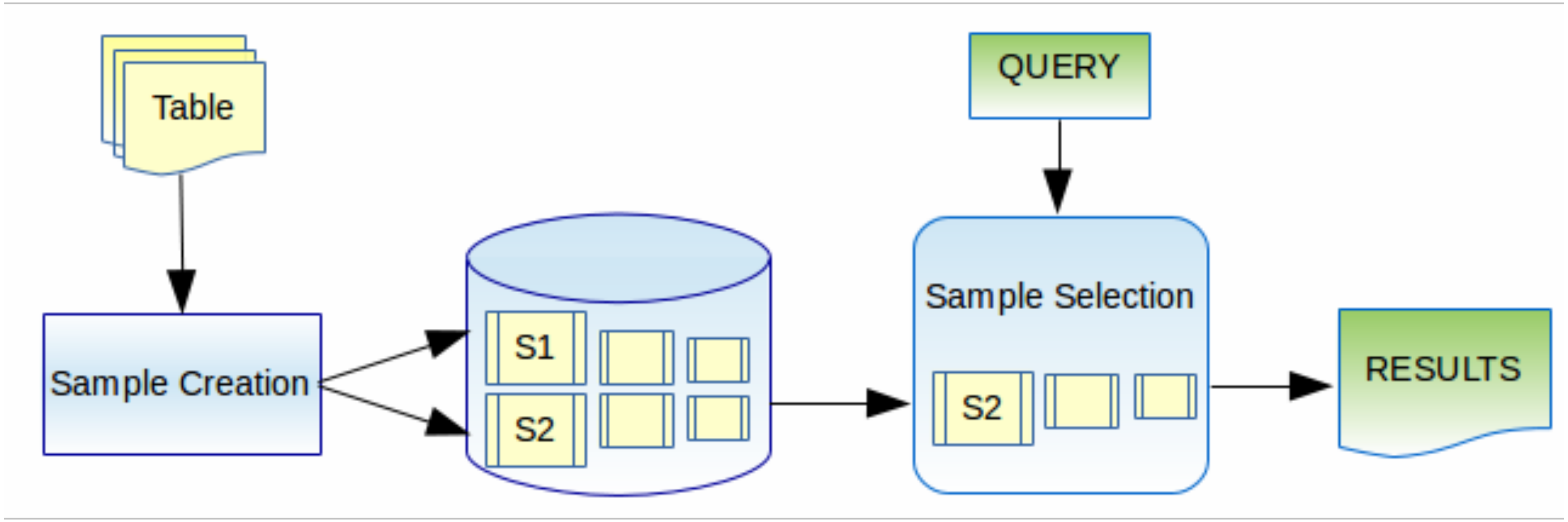
SnappyData



SDE is NOT open source

SnappyData

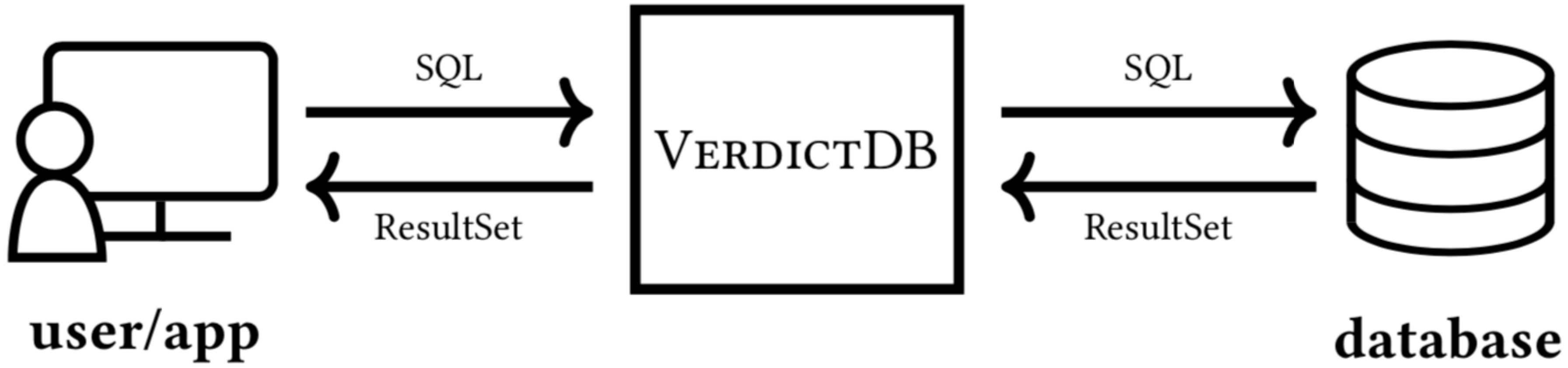
+ WITH ERROR



QCS

FRACTION

VerdictDB



VerdictDB

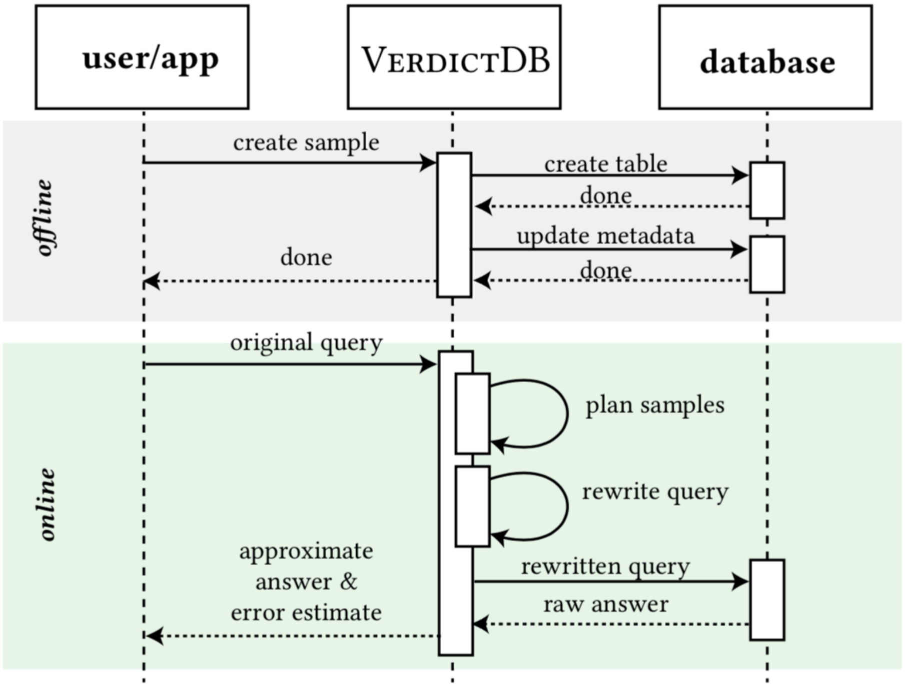
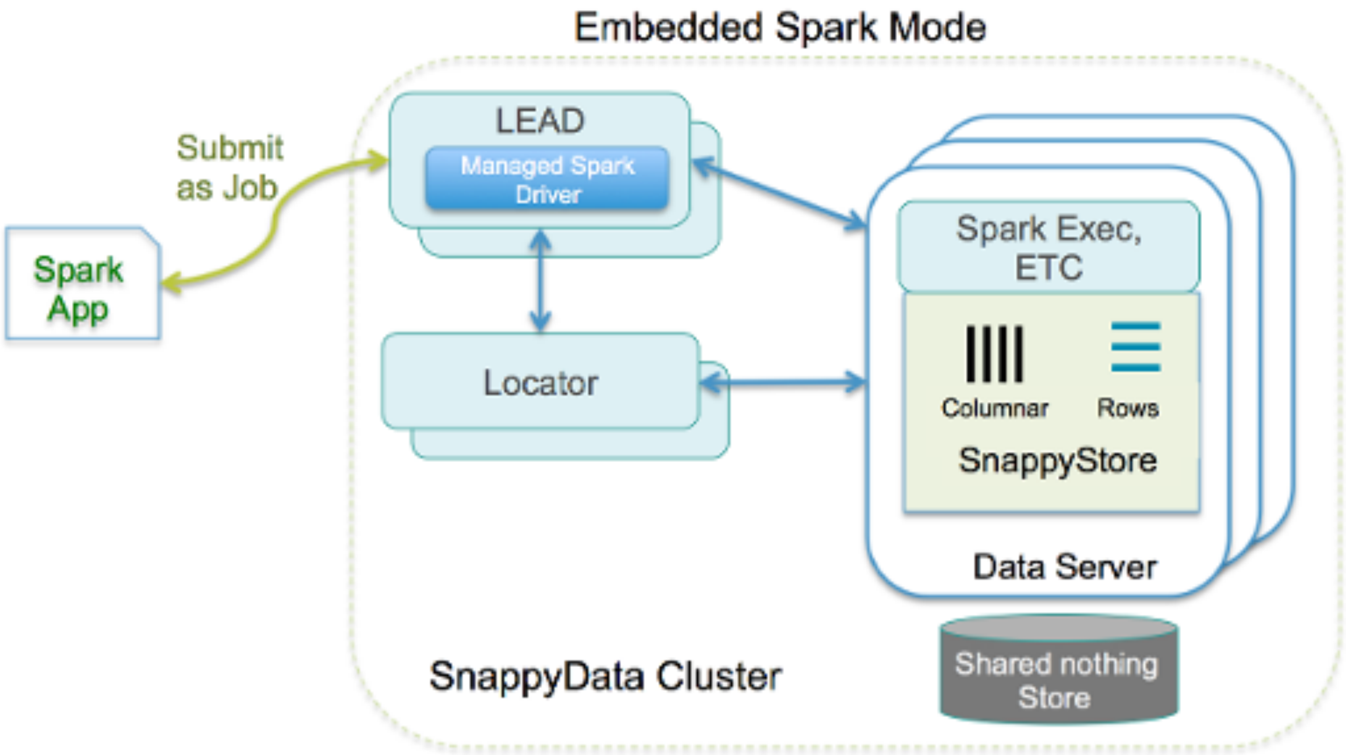


Figure 1: VERDICTDB's offline and online workflow: sample preparation (in gray) and query processing (in green).

Experiment Setup

- Cluster Setup
  - SnappyData: 1 locator, 1 lead, and 2 servers



Experiment Setup

- Cluster Setup
  - SnappyData: 1 locator, 1 lead, and 2 servers
  - VerdictDB on Spark: 1 master and 2 executors
- Each Node
  - 24/32 GB memory used
  - 500 GB HDD

Experiment Setup

- TPC-H Benchmark
  - OLAP
  - 22 queries includes Aggregation, Join, etc.
  - Well known and standard
  - Customizable
- Data
  - 1GB and 10GB
  - Uniformly distributed

Evaluation

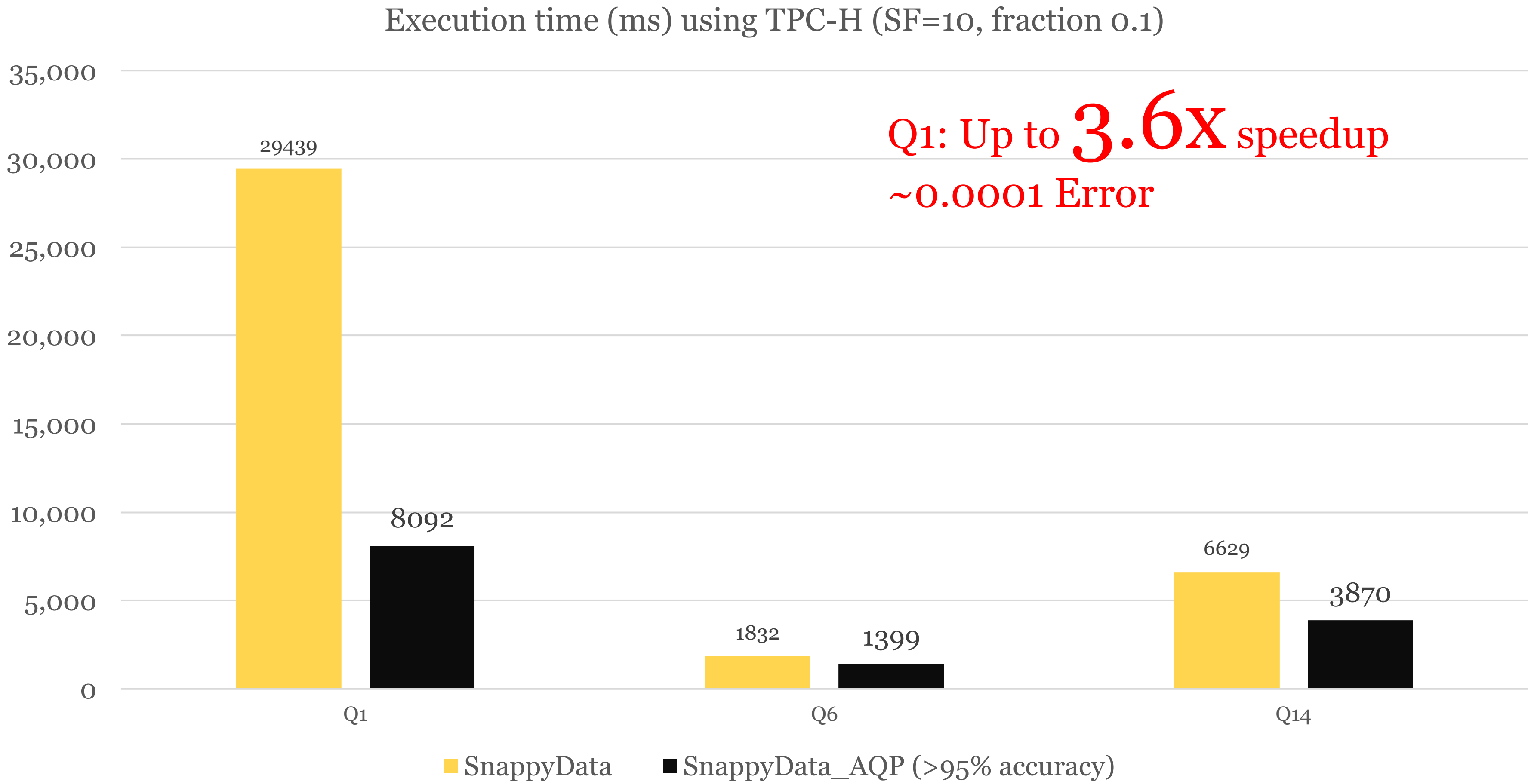
SnappyData

- Stratified Sampling
- In-memory

VerdictDB

- Uniform Sampling
- Not in-memory (bug?)

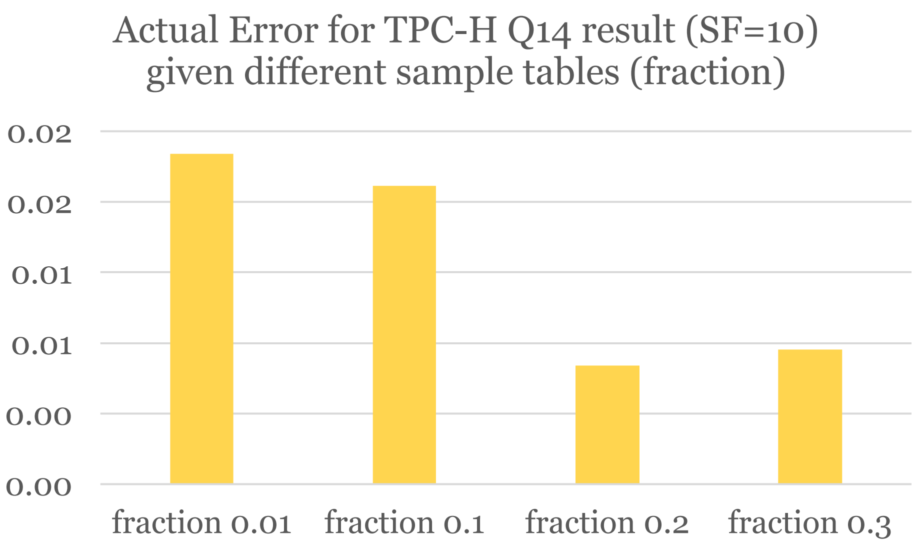
SnappyData - Latency

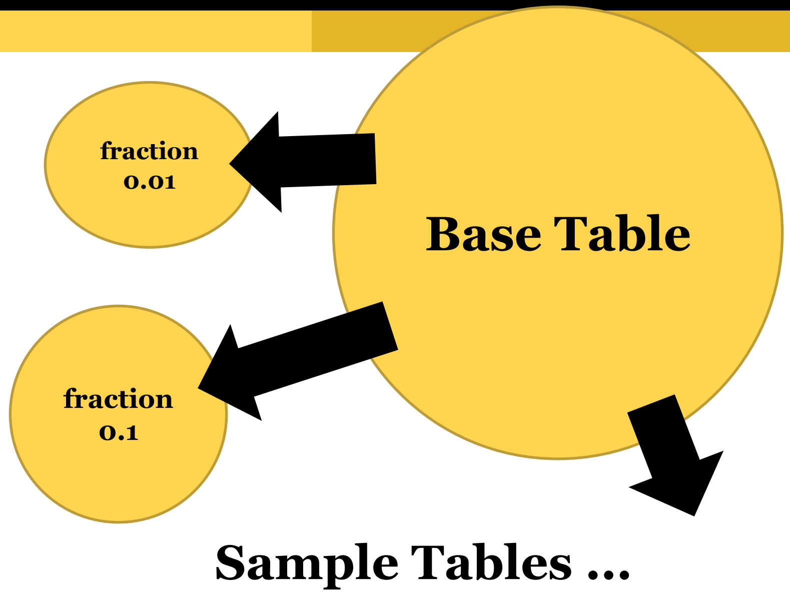


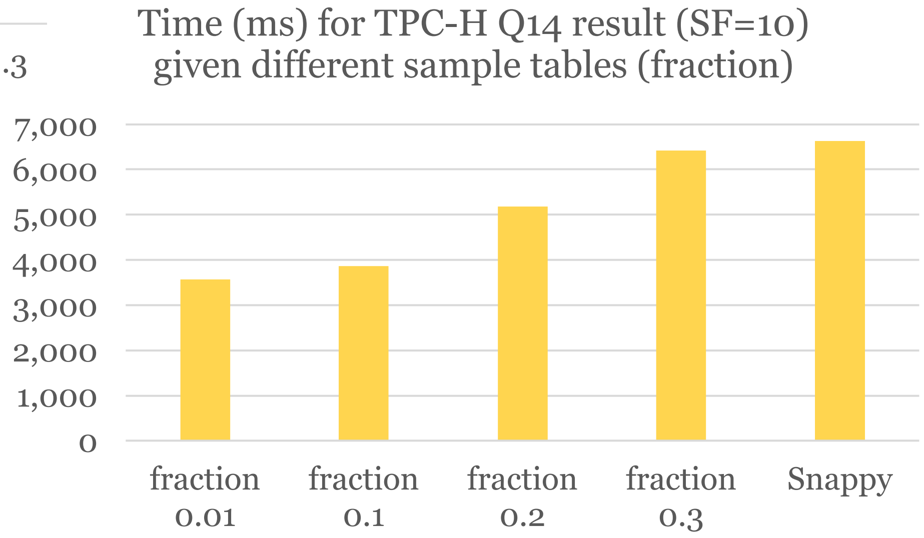
Q1: Up to 3.6x speedup

~0.0001 Error

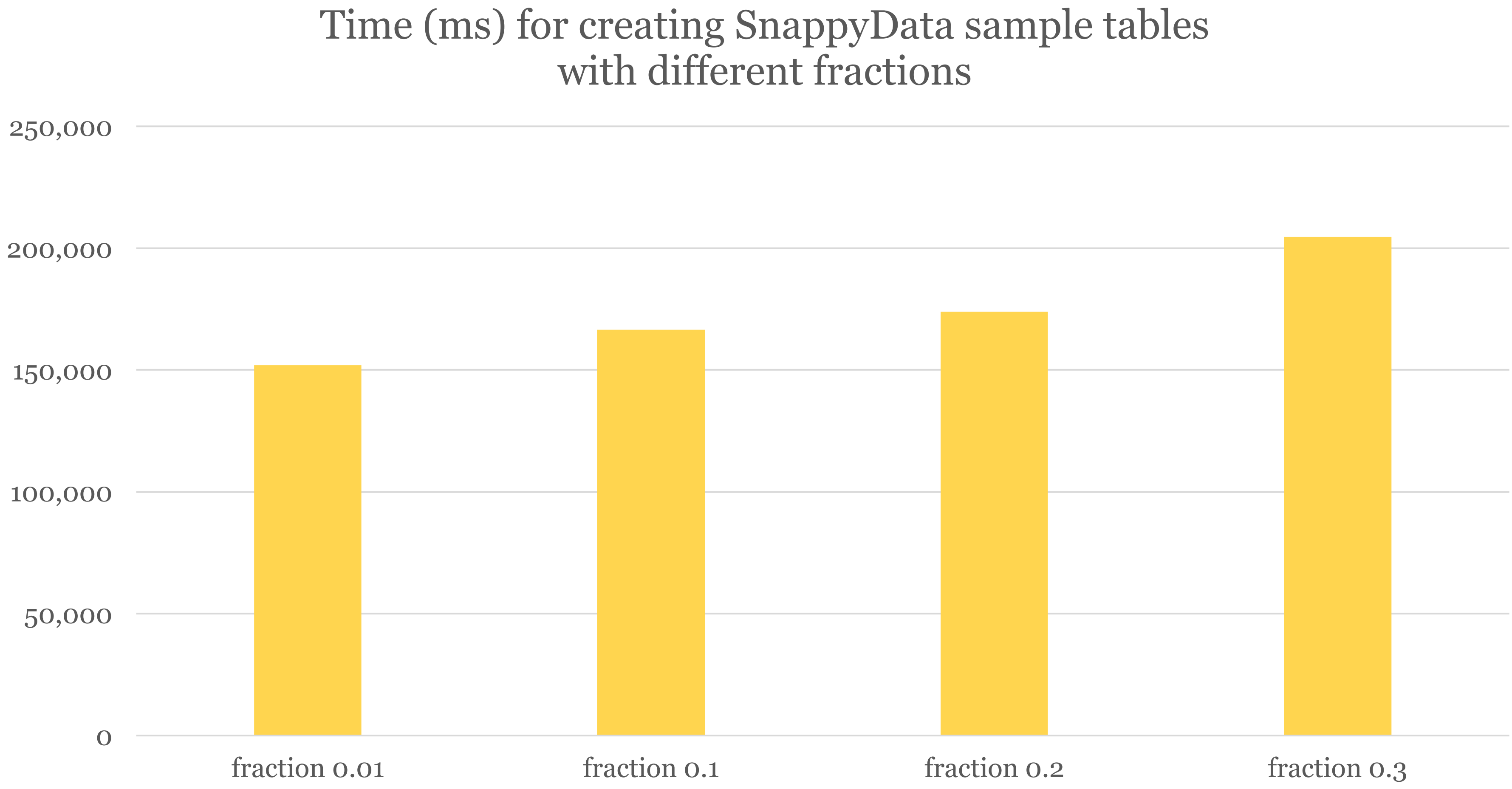
SnappyData - Accuracy



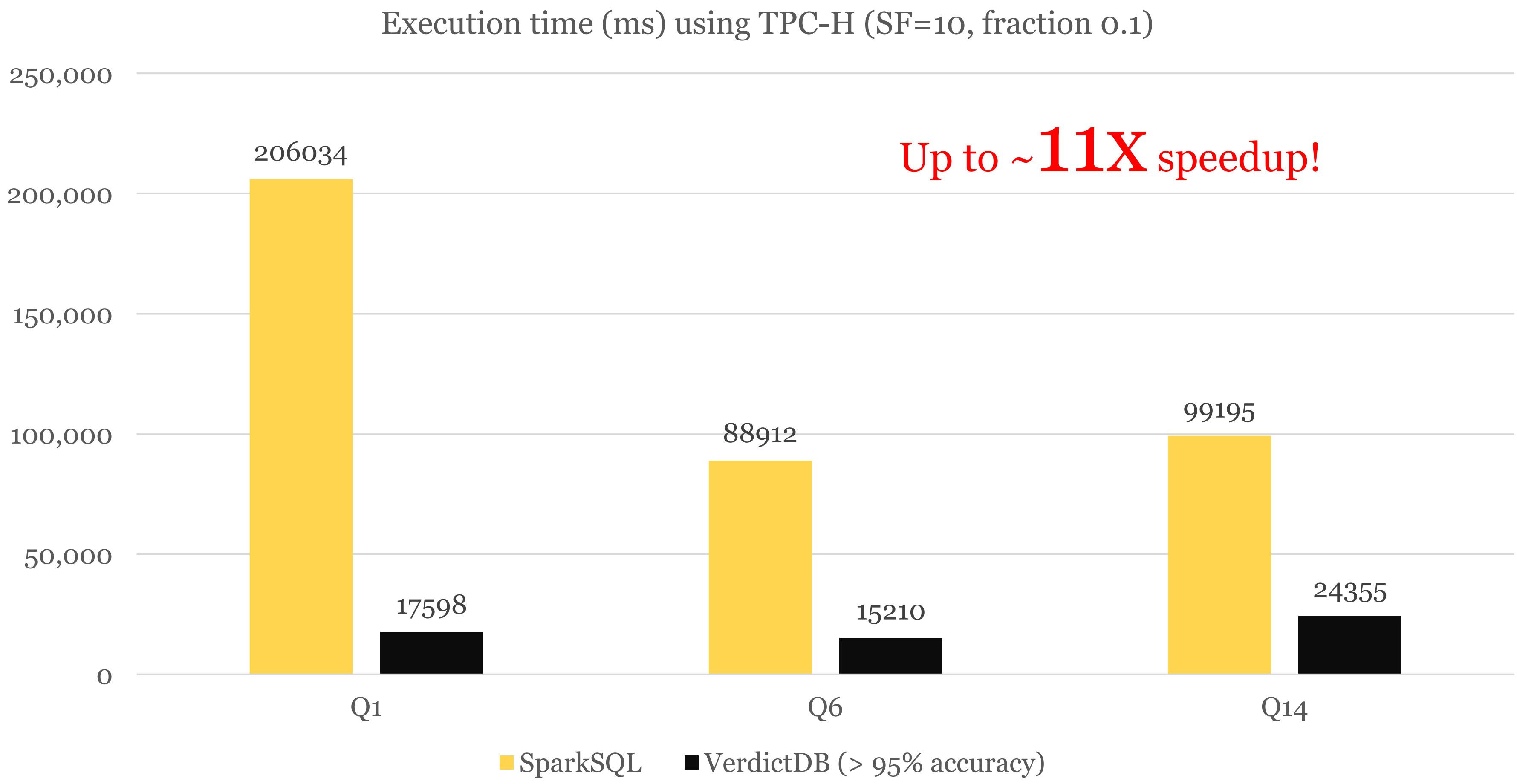




SnappyData- Creating Sample Tables

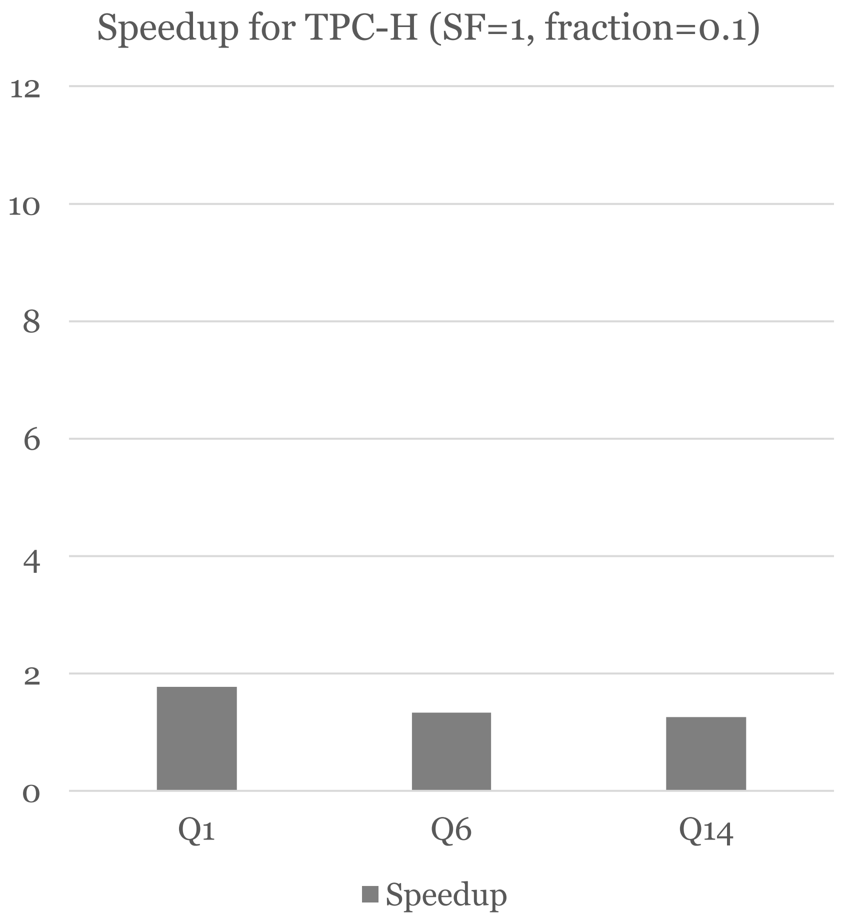


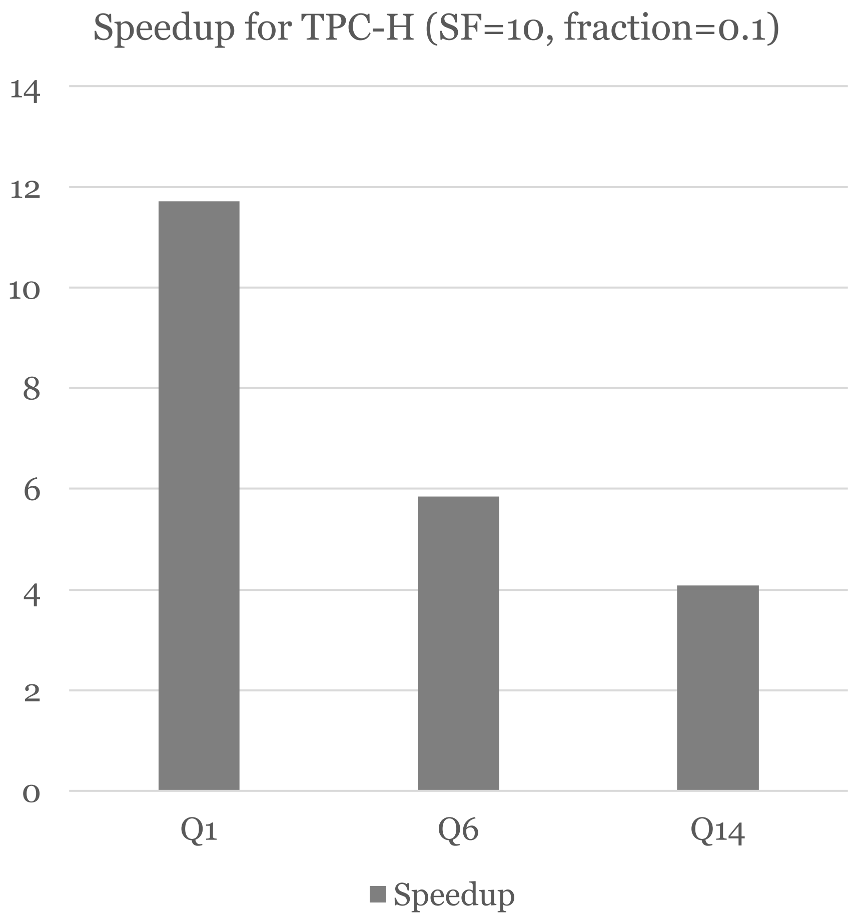
VerdictDB - Latency



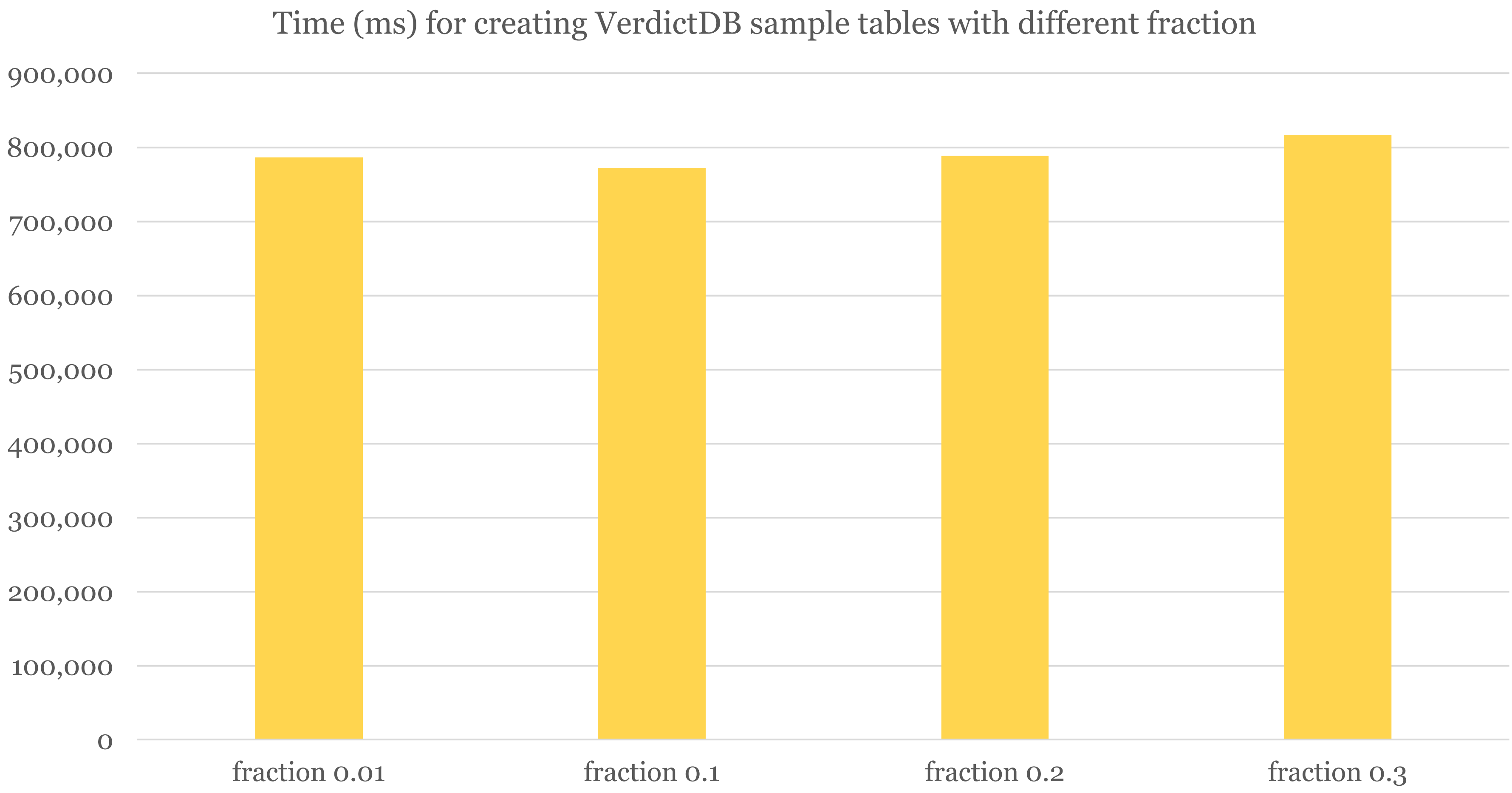
Up to ~11X speedup!

VerdictDB - Speedup

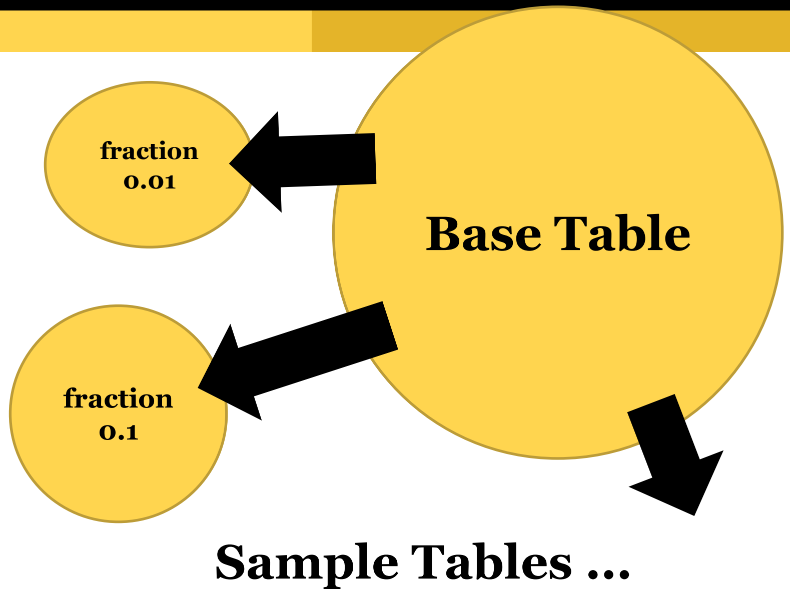




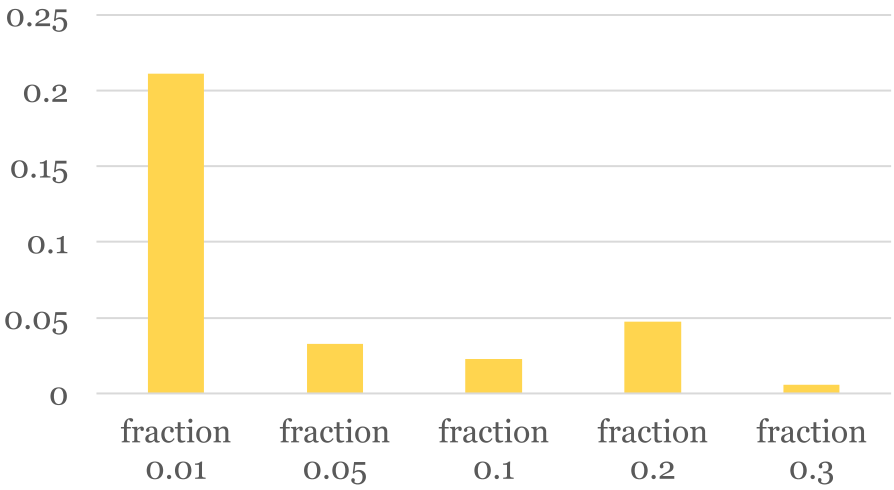
VerdictDB - Creating Sample Tables

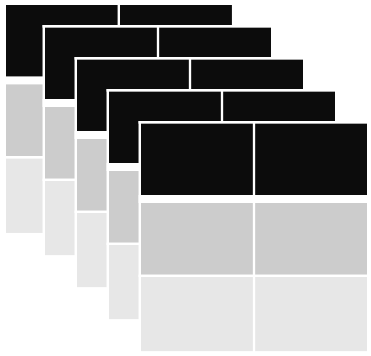


VerdictDB - Accuracy



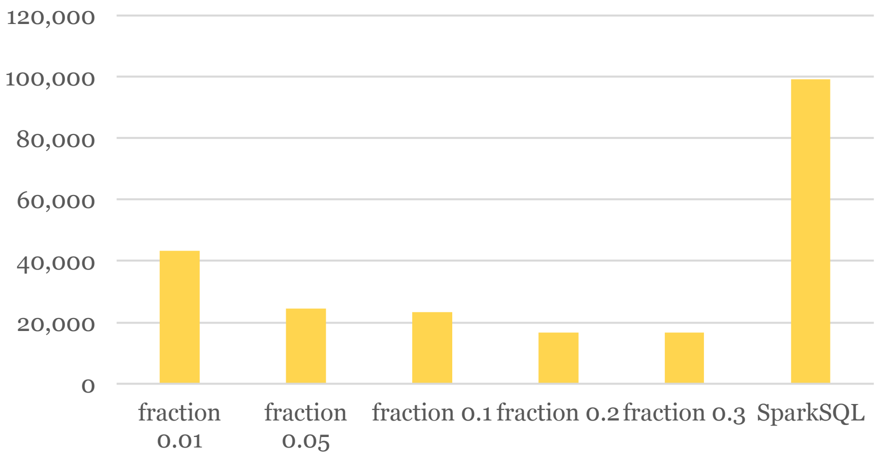
Actual Error for TPC-H Q14 result (SF=10) given different sample tables (fraction)





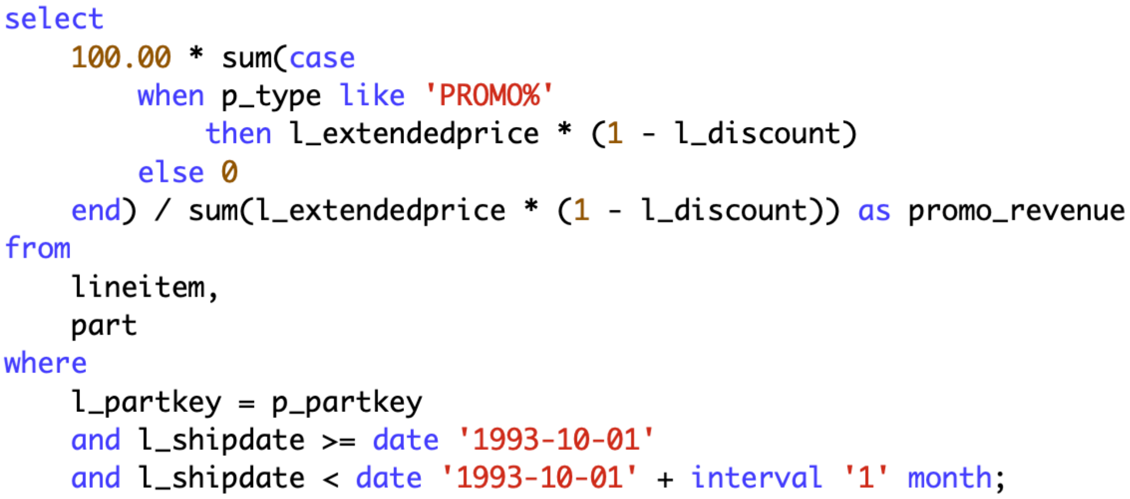
converge!

Time (ms) for TPC-H Q14 result (SF=10) given different sample tables (fraction)

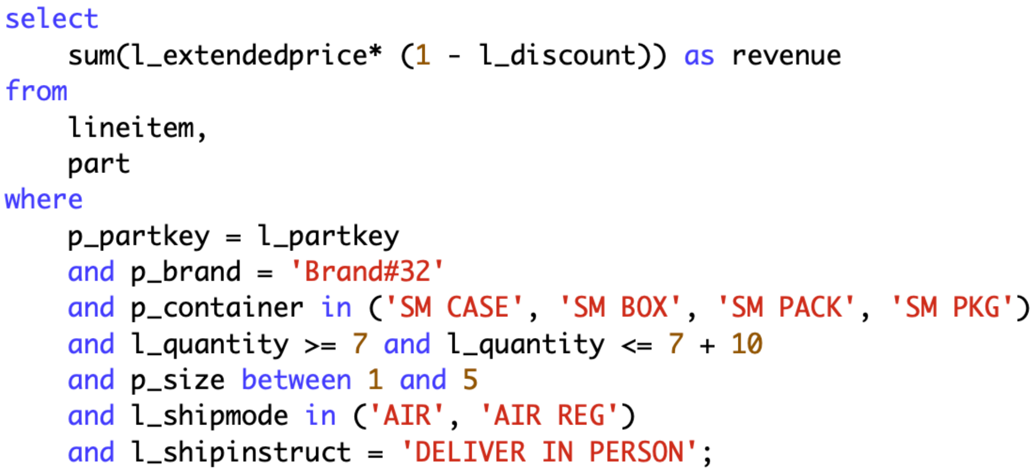


Other Queries?

Q14
Error: ~ 1.7%
Speedup: ~1.7X



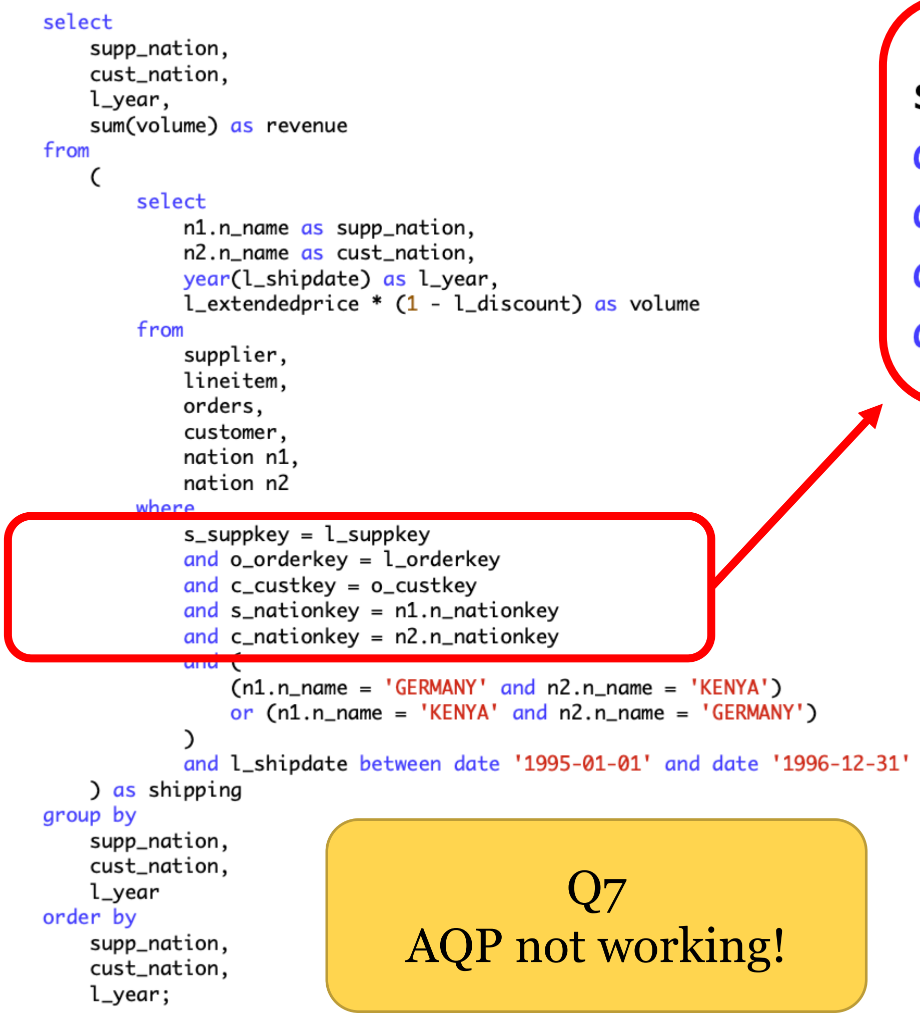
```
select
    100.00 * sum(case
        when p_type like 'PROMO%'
            then l_extendedprice * (1 - l_discount)
        else 0
    end) / sum(l_extendedprice * (1 - l_discount)) as promo_revenue
from
    lineitem,
    part
where
    l_partkey = p_partkey
    and l_shipdate >= date '1993-10-01'
    and l_shipdate < date '1993-10-01' + interval '1' month;
```



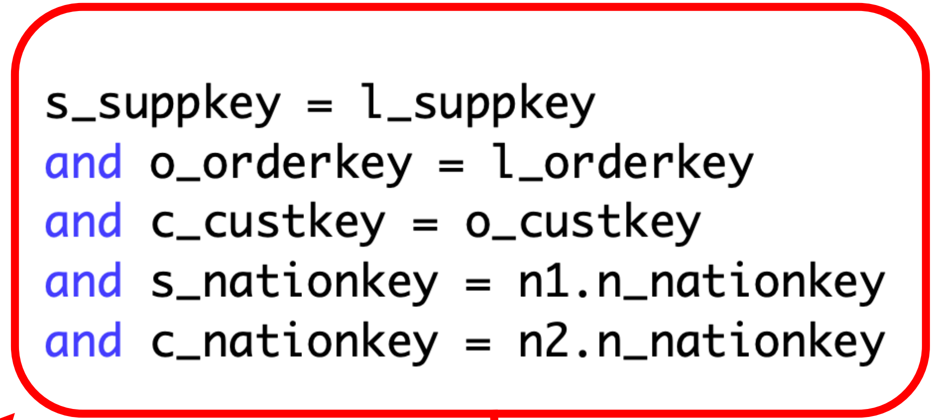
```
select
    sum(l_extendedprice* (1 - l_discount)) as revenue
from
    lineitem,
    part
where
    p_partkey = l_partkey
    and p_brand = 'Brand#32'
    and p_container in ('SM CASE', 'SM BOX', 'SM PACK', 'SM PKG')
    and l_quantity >= 7 and l_quantity <= 7 + 10
    and p_size between 1 and 5
    and l_shipmode in ('AIR', 'AIR REG')
    and l_shipinstruct = 'DELIVER IN PERSON';
```

Q19
Error: ~ 80%
Speedup: ~5.5X

Other Queries?



```
select
    supp_nation,
    cust_nation,
    l_year,
    sum(volume) as revenue
from
    (
        select
            n1.n_name as supp_nation,
            n2.n_name as cust_nation,
            year(l_shipdate) as l_year,
            l_extendedprice * (1 - l_discount) as volume
        from
            supplier,
            lineitem,
            orders,
            customer,
            nation n1,
            nation n2
        where
            s_suppkey = l_suppkey
            and o_orderkey = l_orderkey
            and c_custkey = o_custkey
            and s_nationkey = n1.n_nationkey
            and c_nationkey = n2.n_nationkey
            and (
                (n1.n_name = 'GERMANY' and n2.n_name = 'KENYA')
                or (n1.n_name = 'KENYA' and n2.n_name = 'GERMANY')
            )
            and l_shipdate between date '1995-01-01' and date '1996-12-31'
    ) as shipping
group by
    supp_nation,
    cust_nation,
    l_year
order by
    supp_nation,
    cust_nation,
    l_year;
```



```
s_suppkey = l_suppkey
and o_orderkey = l_orderkey
and c_custkey = o_custkey
and s_nationkey = n1.n_nationkey
and c_nationkey = n2.n_nationkey
```

Key missing in sample tables!

Careful design of sample table or original table!

Q7
AQP not working!

Insights

- AQP performs well:
  - For aggregate functions such as SUM, AVG and COUNT
  - When WHERE is simple
- Users’ foreseen is important!
  - for both query and original table

Future Work

- Test error estimation in sampling
- Other sampling techniques
  - Biased Sampling
- Database learning
- Approximate hardware