Name

CWID

Final Exam

May 6th, 2016
10:30-12:30

CS520 - Data Integration, Warehousing, and Provenance

Results

Please leave this empty!

1.1 1.2 1.3

Sum

Instructions

- Try to answer all the questions using what you have learned in class. Keep hard questions until the end.
- **When writing a query, write the query in a way that it would work over all possible database instances and not just for the given example instance!**
- The exam is closed book and closed notes! No calculator, smartphones, or similar allowed!

Consider the following datawarehouse schema (star schema) and partial example instance. There is a single fact table about calls and four dimension tables for the following dimensions:

- **Time** with four levels (year, month, day, hour)
- **Location** with three levels (state, city, zip)
- **Customer** with two levels (type, SSN)
- **Provider** with two levels (network, providerName)

callfacts

LID	TID	CID	PID	numCalls
1	4	1	1	15
2	1	5	2	10
100	1	76	4	22
...	...	...	...	...

customerDim

CID	type	SSN
1	premium	122-324-3454
1	standard	222-324-3454
...	...	...

timeDim

TID	year	month	day	hour
1	2015	1	1	1
2	2015	1	1	2
...	...	...	...	...
...	2016	5	6	23

locationDim

LID	state	city	zip
1	Illinois	Chicago	60616
2	Lousiana	New Orleans	42345
...	...	...	...

providerDim

PID	network	providerName
1	CDMA	Sprint
2	CDMA	Verizon
3	GSM	AT&T
4	GSM	T-Mobile

Hints:

- Attributes with black background form the primary key of a relation (e.g., `PID` for relation `providerDim`)
- Attributes `LID`, `TID`, `PID`, and `CID` in the fact table are foreign keys to the dimension tables

Part 1.1 Data Warehousing (Total: 40 Points)

Recall that you should write all queries according to the schema and not according to the example instance.

Question 1.1.1 (6 Points)

Write an SQL query that returns the total number of calls in the database and also the number of calls per year, per month, and per day.

Solution

```
SELECT sum(numCalls)
FROM callfacts f NATURAL JOIN timeDim t
GROUP BY ROLLUP (year, month, day);
```

Question 1.1.2 (6 Points)

Write an SQL query that returns the total number of calls made by premium customers on December 31st (month 12, day 31) of 2015 in Chicago.

Solution

```
SELECT sum(numCalls)
FROM callfacts f
     NATURAL JOIN
     (SELECT CID FROM customerDim WHERE type = 'premium') c
     NATURAL JOIN
     (SELECT TID FROM timeDime WHERE year = 2015 AND month = 12 AND day = 31) t
     NATURAL JOIN
     (SELECT LID FROM locationDim WHERE city = Chicago) l
```

Question 1.1.3 (7 Points)

Write an SQL query that returns year and month pairs during which no calls where made using the *Sprint* provider. For example, if there where no Sprint calls made in January 2016 then `(2016,1)` should be in the result.

Solution

```
SELECT DISTINCT year, month
FROM timeDim t1
WHERE NOT EXISTS (SELECT *
                  FROM callfacts f NATURAL JOIN timeDim t2
                  WHERE t2.year = t1.year AND t2.month = t1.month)
```

if students assume that every time unit has an entry then this is also correct:

```
SELECT DISTINCT year, month
FROM timeDim t1
WHERE NOT EXISTS (SELECT *
                  FROM callfacts f NATURAL JOIN timeDim t2
                  WHERE t2.year = t1.year AND t2.month = t1.month AND numCalls != 0)
```

Question 1.1.4 (7 Points)

Write an SQL query that returns the name of the provider with the most calls.

Solution

```
SELECT providerName
FROM (SELECT providerName,
             sum(numCalls) OVER (PARTITION BY providerName) AS ttl
      FROM callfacts f NATURAL JOIN providerDim p)
ORDER BY ttl DESC
LIMIT 1
```

Question 1.1.5 (7 Points)

Write an SQL query that returns for each provider a rolling sum for the total number of calls per month in 2015. For example, if there where 20 Sprint calls in January 2015 and 30 Spring calls in February 2015 then the rolling sum for January would be 20 and the one for February would be 50.

Solution

```
SELECT DISTINCT providerName, month,
                sum(numCalls) OVER (PARTITION BY providerName ORDER BY month)
FROM callfacts f NATURAL JOIN providerDim p NATURAL JOIN timeDime t
WHERE year = 2015;
```

Question 1.1.6 (7 Points)

Write an SQL query that returns for each provider and year the name of the city with the most and the name of the city with the least amount of calls. For example, a result for Sprint and 2015 may be `(Sprint, 2015, Chicago, New York)` if Chicago had the most calls on the Sprint provider network during 2015 and New York had the least.

Solution

```
SELECT DISTINCT providerName, year,
       first_value(city) OVER (PARTITION BY providerName, year
                               ORDER BY ttl
                               ROWS UNBOUNDED PRECEDING),
       last_value(city) OVER (PARTITION BY providerName, year
                              ORDER BY ttl
                              ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING)
FROM (SELECT providerName, year, city, sum(numCalls) AS ttl
      FROM callfacts f
           NATURAL JOIN timeDim t
           NATURAL JOIN locationDim l
           NATURAL JOIN providerDim p
      GROUP BY providerName, year, city) sub
```

Part 1.2 Big Data (Total: 30 Points)

Question 1.2.1 Basic Concepts (2.5 Points)

- ❑ Any algorithm can be efficiently parallelized and distributed
- ■ The pay-as-you-go integration approach can lead to repeated efforts (e.g., an input dataset is cleaned once for each analysis)
- ❑ The pay-as-you-go integration approach has the advantage that data can be processed in a more timely fashion because no schema and ETL process needs to be designed upfront
- ■ Load balancing ensures that computational load is distributed evenly across machines in a cluster
- ❑ Fault tolerance is more important for smaller clusters than for larger clusters

Question 1.2.2 Distributed file systems (2.5 Points)

- ■ In HDFS there is a single node storing the file system metadata, but many nodes storing the data (file content)
- ❑ HDFS provides a random access interface for writing to an existing file, i.e., we can overwrite any part of an existing file in HDFS
- ■ Fault tolerance in HDFS is achieved through replication
- ❑ Reads in HDFS are slower than writes
- ❑ In HDFS, clients only communicate with the name node, but never with any of the data nodes

Question 1.2.3 HDFS Block Size (4 Points)

Name two reasons for why HDFS only works well with large block sizes (e.g., 128MB).

Solution

- name node is bottleneck for metadata access and small blocks would result in large amounts of metadata
- sequential access to files for speed

Question 1.2.4 HDFS Fault Tolerance (4 Points)

How does HDFS detect that a data node has crashed? Just name the mechanism that is used or describe it in 1-2 sentences.

Solution

- heartbeat messages that are send from data nodes to the name node

Question 1.2.5 MapReduce and Hadoop (4 Points)

- ❑ In Hadoop, after each map phase the results are written to HDFS
- ❑ The shuffle operation does not require any communication among the worker nodes
- ■ The combiner operation can improve performance of a MR computation
- ❑ Data locality is not taken into account when a Map job is scheduled
- ❑ The user provided map function takes as input a key and a list of values
- ■ Map and Reduce are higher-level functions that take as input a collection and a user defined function
- ❑ More complex computations in MR can be written by connecting multiple MR jobs into a more complex workflow
- ❑ No more than one map job will run on each node of a Hadoop cluster

Question 1.2.6 (4 Points)

Describe in 2-3 sentences how Hadoop achieves fault tolerance.

Solution

Question 1.2.7 (4 Points)

Describe in 2-3 sentences how Spark achieves fault tolerance.

Solution

Question 1.2.8 (5 Points)

When should we use a broadcast join and when a reduce join (partition join). Describe in 2-3 sentences.

Solution

Part 1.3 Provenance (Total: 30 Points)

For the following the queries over the schema shown below, compute the provenance according to the following provenance models for all their result tuples.

- Why-Provenance
- Minimal Why-Provenance
- Provenance Polynomials

Before presenting provenance, show the results for each query first and label the result tuples $t_1, t_2, \ldots, t_n$.

Consider the following database schema and instance:

location

lName	city	owner	sizeSf	
Windsor Castle	Windsor	Queen	40,000	l_1
Big Ben	London	Public	3,500	l_2
Stonehedge	Amesbury	Public	14,000	l_3

account

witness	suspect	crimeId	
Bob	Peter	1	a_1
Peter	Bob	1	a_2
Queen	Bob	2	a_3

crime

crimeId	lName	time	type	victim	
1	Big Ben	10:30	murder	Alice	c_1
2	Windsor Castle	11:00	theft	Queen	c_2

Question 1.3.1 (5 Points)

$$\pi_{suspect}(account)$$

Solution

Result relation:

suspect	
Peter	t_1
Bob	t_2

Why provenance:

name	
Peter	$\{\{a_1\}\}$
Bob	$\{\{a_2\},\{a_3\}\}$

Minimal Why provenance:

name	
Peter	$\{\{a_1\}\}$
Bob	$\{\{a_2\},\{a_3\}\}$

Provenance Polynomials:

name	
Peter	a_1
Bob	$a_2 + a_3$

Question 1.3.2 (8 Points)

$$q = \pi_{time,type,victim}(crime \rtimes \sigma_{city=Windsor}(location))$$

Solution

Result relation:

time	type	victim	
11:00	theft	Queen	t_1

Why provenance:

time	type	victim	
11:00	theft	Queen	$\{\{l_1, c_2\}\}$

Minimal Why provenance:

time	type	victim	
11:00	theft	Queen	$\{\{l_1, c_2\}\}$

Provenance Polynomials:

time	type	victim	
11:00	theft	Queen	$l_1 \times c_2$

Question 1.3.3 (8 Points)

$$q = \rho_{p \leftarrow witness}(\pi_{witness}(account) \cup \pi_{suspect}(account) \cup \pi_{victim}(crime))$$

Solution

Result relation:

p	
Bob	t_1
Peter	t_2
Queen	t_3
Alice	t_4

Why provenance:

p	
Bob	$\{\{a_1\},\{a_2\},\{a_3\}\}$
Peter	$\{\{a_1\},\{a_2\}\}$
Queen	$\{\{a_3\},\{c_2\}\}$
Alice	$\{\{c_1\}\}$

Minimal Why provenance:

p	
Bob	$\{\{a_1\},\{a_2\},\{a_3\}\}$
Peter	$\{\{a_1\},\{a_2\}\}$
Queen	$\{\{a_3\},\{c_2\}\}$
Alice	$\{\{c_1\}\}$

Provenance Polynomials:

name p	
Bob	$a_1 + a_2 + a_3$
Peter	$a_1 + a_2$
Queen	$a_3 + c_2$
Alice	c_1

Question 1.3.4 (9 Points)

$$q = \rho_{name \leftarrow witness}(\pi_{witness}(\sigma_{suspect=witness}((\pi_{witness,crimeId}(account) \bowtie \pi_{suspect,crimeId}(account)))))$$

Solution

Result relation:

name	
Bob	t_1
Peter	t_2

Why provenance:

name	
Bob	$\{\{a_1, a_2\}\}$
Peter	$\{\{a_1, a_2\}\}$

Minimal Why provenance:

name	
Bob	$\{\{a_1, a_2\}\}$
Peter	$\{\{a_1, a_2\}\}$

Provenance Polynomials:

name	
Bob	$a_1 \times a_2$
Peter	$a_1 \times a_2$