CS-5513 Advance Database Management

Homework 1

Submitted By

Shyam Sundar Murali Krishnan

shyamkrishnan@ou.edu

113475983

Problem 1:

a) A customer submits a query to the Baker Movies LA site to find the average ticket cost at each movie theater.

Since the customer submits query to the LA site to find the average ticket cost for each movie theatre, we could directly access the MovieShowsIn table which consists of the tickets cost and has the corresponding theatre name to it. So, we could locally access this table since this table is present on the Baker Movies LA site itself to fetch the results. The corresponding query can be answered by averaging the ticket cost for every group of theatre name.

${}_{\text{theater\_name}}G_{\text{avg(ticket\_cost)}}(\text{MovieShowsIn})$

b) A manager at Baker Movies SF submits a query to this site to find the names of the employees working at the movie theaters that are showing the movie titled “Spider-Man: No Way Home.”

Since the baker movies site SF has the Movietheater relation and few tuples of Employee with respect to the theater name Baker Movies SF. Now
temp_1 -> $\Pi_{\text{name, theater\_name}}$ (Employee)

now this temp1 is shipped to baker movies LA site. In the LA site the following operations are done.
temp_2 -> $\sigma_{\text{movie\_title = “Spider-Man: No Way Home”}}(\text{MovieShowsIn})$

temp_3 -> $\Pi_{\text{Employee.name}}$ (Employee ⋈ temp_2)
temp_1 -> $\Pi_{\text{temp\_1.name}}$ (temp_1 ⋈ temp_2)

Now the temp_2 is shipped to baker movies NYC site
temp_2 -> $\Pi_{\text{Employee.name}}$ (Employee ⋈ temp_2)

Now the temp_2 is shipped back to baker movies LA site.
temp_1 -> temp_1 U temp_2
temp_1 -> temp_1 U temp_3

Now the temp_1 containing the names of the employees working at the movie theaters that are showing the movie titled “Spider-Man: No Way Home” is shipped back to baker movies SF site.

Problem 2:

a) Let us consider the database name to be “customerproduct”.

```
connect customerproduct@ou.edu
Set transaction Name=’Average_quantity’
Start transaction
Begin:

SELECT T.cust_id as cust_id, B.quantity as quantity INTO temp_table FROM
gpel9.BoughtInTransaction@customerproduct.ou.edu as B JOIN
gpel9.Transaction@customerproduct.ou.edu as T USING (trans_id);

SELECT c.name, avg(T.quantity) as average FROM
gpel7.Customer@customerproduct.ou.edu as c JOIN temp_table as T USING (cust_id) GROUP BY
c.name;

End

Commit;
```

References:

1. https://docs.oracle.com/en/database/oracle/oracle-database/21/admin/managing-distributed-transactions.html#GUID-45329B02-514E-446E-89FE-14BA008BEA07
2. https://docs.oracle.com/database/121/ADMIN/ds_concepts.htm#ADMIN02803

b) Each database instance involved in a two-phase commit operation. There are three steps in two-Phase Commit processing:

a) Prepare
b) Commit
c) Forget

a) Prepare phase:
The first phase in committing a distributed transaction is the prepare phase. In this phase, Oracle does not actually commit or roll back the transaction. Instead, all nodes referenced in a distributed transaction are told to prepare to commit. The types of response got from the response phase is

1. Prepared: Data on the node has been modified by a statement in the distributed transaction, and the node has successfully prepared.
2. Read-only: No data on the node has been, or can be, modified, so no preparation is necessary.
3. Abort: The node cannot prepare.

b) Commit Phase:
The second phase in committing a distributed transaction is the commit phase. Before this phase occurs, *all* nodes other than the commit point site (Node which is first to commit as part of the two-phase commit process) referenced in the distributed transaction have guaranteed that they are prepared have the necessary resources to commit the transaction.

c) Forget Phase:
After the participating nodes notify the commit point site that they have committed, the commit point site can forget about the transaction.

References:

3.https://docs.oracle.com/cd/B10500_01/server.920/a96521/ds_txns.htm

4.https://www.doag.org/formes/pubfiles/6380784/2014-DB-Joel_Goodman-Oracle_Distributed_Transactions_-Praesentation.pdf

Problem 3:

a) Assume that American Airlines have three main offices around the US. A main office is located for every time zone. There are three main time zones in US namely the eastern, central, and pacific time zones. For the eastern time zone, the main office is situated in Tampa, for the pacific time zone the main office is in San Jose and for central time zone the main office is in Illinois. Each main office is the node of a distributed database. Also assume that the eastern time zone office is the oldest office and is currently under DB maintenance and their systems are getting upgraded.

The following relations are:

Airport (a.name, Zone, location)

Employee (e.id, Name, salary, zone)

Flight (f.no, arr_time, no_stops, depart_time, arr_airport)

Departsfrom (f.no, a.name, zone)

The queries are:

1. A manager of central zone main office queries to collect the average salary of the employees across the time zones. (1/week)
2. A customer from the eastern time zone queries to know the flight number which has stops less than 2 and arrives to “Ohare international airport”. (100/minute)
3. A customer from the pacific time zone queries to get the flight details that departs from “Los Angeles international airport” to “Texas Dallas international airport” (100/hour).

b) The flight relation is replicated into Tampa office site and Illinois office site. This is because as mentioned above the Tampa office is currently under maintenance and since this is the oldest office the main relation flight must be in Tampa office site. But due to currently undergoing maintenance situation it will be safer to replicate to another site which is in this case Illinois office site.

c) The Employee relation is fragmented horizontally based on the time zone and its details are kept at each site. It is fragmented in this way because it will be wrong if a single site can access the details of employees from other office site as well. That’s why it is fragmented according to its office site which is their respective time zone.

The departurefrom and the airport relations are stored in San Jose office site.

d) A manager of central zone main office queries to collect the average salary of the employees across the time zones. (1/week)
temp_1 -> $\Pi_{salary,\ zone}$ (Employee)

Now the temp_1 is shipped to Tampa office site.
temp_2 -> $\Pi_{salary,\ zone}$ (Employee)
temp_1 -> temp_1 U temp_2

Now the temp_1 is shipped to San Jose office site.
temp_2 -> $\Pi_{salary,\ zone}$ (Employee)
temp_1 -> temp_1 U temp_2
temp_1 -> ${}_{zone}G_{avg(salary)}$(temp_1)

Now the temp_1 is shipped back to Illinois office site and the answer is given to the manager.

A customer from the eastern time zone queries to know the flight number which has stops less than 2 and arrives to “Ohare international airport”. (100/minute)

Since the relation is already there in the local site it can be directly accessed and provide the answer by

Π_{f_no} ($\sigma_{\text{arr\_airport = “Ohare international airport” and no\_stops <2}}$(Flight))

A customer from the pacific time zone queries to get the flight details that departs from “Los Angeles international airport” to “Texas Dallas international airport” (100/hour).

The pacific zone site has the departurefrom relation. So

temp_1 -> $\Pi_{f\_no,\ zone}$ ($\sigma_{\text{a\_name = “Los Angeles international airport”}}$ (departurefrom))

Now this temp_1 is sent to central time zone office at Illinois.

temp_1 ->$\sigma_{\text{Flight.arr\_airport = “Texas Dallas international airport”}}$ (Flight $\bowtie$ temp_1)

Now this temp_1 is sent back to the pacific zone site and is given as the answer.

Bonus Problem:

Locks are used for two main purposes in Oracle multiuser management is:

1. to provide transaction isolation
2. to provide cache coherency.

Local Locks:

Locking mechanisms which are synchronized within a single instance. These are used in Oracle in both exclusive and shared modes.

Enqueues:

Enqueues are shared memory structures which serialize access to resources in the database. These locks can be local to one instance or global to a database. They are associated with a session or transaction.

Instance Locks:

Instance locks only come into existence if you start an Oracle instance in shared mode. They synchronize between instances, communicating the status of a resource among the instances of an Oracle multiuser management. It is of two types PCM lock and non-PCM locks

PCM locks:

Parallel cache management locks cover one or more data blocks in the buffer. PCM locks do not lock any rows on behalf of transactions.

Non- PCM locks:

Non-PCM locks of many different kinds, control access to data and control files and perform various types of communication between instances.

Reference:

5.https://docs.oracle.com/cd/A57673_01/DOC/server/doc/SPS73/chap7.htm