

Tutorial 3: Distributed Transactions and System Integration

Semester 1, 2004

Question 1: Implement the Basic Timestamp Ordering Algorithm and determine how these schedules will complete.

The initial timestamp values are

- read-TS (X) = 1.0
- read-TS (Y) = 1.2
- read-TS (Z) = 2.2
- write-TS (X) = 1.4
- write-TS (Y) = 1.6
- write-TS (Z) = 2.4

The timestamps of the transactions are

- TS (T1) = 3.0
- TS (T2) = 3.2

Schedule 1

T1	T2
read-item (X);	read-item (Z);
read-item (Y);	Z:= M + N;
Y:= Y + X;	write-item (Z);
write-item (Y);	

Schedule 2

T1	T2
read-item (X);	read-item (X);
	read-item (Y);
	X:= X + Y;
	write-item (X);
X:= X + N;	
write-item (Y);	read-item (Z);
	Z:= Z + Y;
	write-item (Z);

Solutions:

Schedule 1:

All operations are permitted to execute.

T1	T2
read-item (X);	read-item (Z); Z:= M + N; write-item (Z);
read-item (Y);	
Y:= Y + X;	
write-item (Y);	

```
read_TS(X) : 1.0 →3.0
read_TS(Z) : 2.2 →3.2
read_TS(Y) : 1.2 →3.0

write_TS(Z) : 2.4 →3.2

write_TS(Y) : 1.6 →3.0
```

The resulting timestamps are:

```
read-TS (X) = 3.0
read-TS (Y) = 3.0
read-TS (Z) = 3.2
write-TS (X) = 1.4 (unchanged)
write-TS (Y) = 3.0
write-TS (Z) = 3.2
```

Schedule 2:

T1 is aborted at write-item (Y); because the read timestamp of Y (3.2) at that point is greater than the timestamp of T1 (3.0).

T1	T2
read-item (X);	read-item (X); read-item (Y); X:= X + Y; write-item (X);
X:= X + N;	
write-item (Y);	
	read-item (Z); Z:= Z + Y; write-item (Z);

```
read_TS(X) : 1.0 →3.0
read_TS(X) : 3.0 →3.2
read_TS(Y) : 1.2 →3.2

write_TS(X) : 1.4 →3.2

write_TS(Y) : read_TS(Y) > TS(T1), T1 aborts!
read_TS(Z) : 2.2 →3.2

write_TS(Z) : 2.4 →3.2
```

The resulting timestamps are:

```
read-TS (X) = 3.2
read-TS (Y) = 3.2
read-TS (Z) = 3.2
write-TS (X) = 3.2
write-TS (Y) = 1.6 (unchanged)
write-TS (Z) = 3.2
```

Question 2: Consider the following Employees and Departments relations:
 Employees(eid: integer, did: integer, sal: real)

Departments(did: integer, mgrid: integer, budget: integer)

Suppose that the Employees relation is stored in a database in Adelaide and the tuples with $sal \leq 100,000$ are replicated in a database in Sydney. Consider the following three options for lock management: all locks managed at a single site, say, Brisbane; primary copy with Adelaide being the primary for Employees; and fully distributed. For each of the lock management options, explain what locks are set (and at which site) for the following queries. Also state which site the page is read from.

(a) A query submitted in Darwin wants to read a page containing Employees tuples with $sal \leq 50,000$.

(b) A query submitted in Adelaide wants to read a page containing Employees tuples with $sal \leq 50,000$.

Solutions:

(a) Darwin does not have a copy of the relevant partition. So it needs to query either the Sydney or Adelaide databases. In the following table, wherever Sydney and Adelaide are listed together, both possibilities exist; the choice is the city the query is shipped to.

Protocol	Locks Set At	Pages Read From
Single Site	Brisbane	Adelaide/Sydney
Primary Copy	Adelaide	Adelaide/Sydney
Fully Distributed	Adelaide/Sydney	Adelaide/Sydney

In the last case, the locks are set at, and the pages are read from the same site – depending on where the query is shipped.

(b) Adelaide can query its own database. So the query and the result do not have to be shipped.

Protocol	Locks Set At	Pages Read From
Single Site	Brisbane	Adelaide
Primary Copy	Adelaide	Adelaide
Fully Distributed	Adelaide	Adelaide

Question 3:

Suppose that 2PC with Presumed Abort is used as the commit protocol. Explain how the system recovers from failure and deals with particular transaction T in each of the following cases:

- (a) A subordinate site for T fails before receiving a *prepare* message.
- (b) A subordinate site for T fails after receiving a *prepare* message but before making a decision.
- (c) The coordinator site for T fails before sending a *prepare* message.

- (d) The coordinator site for T fails after writing an *abort* log record but before sending any further messages to its subordinates.

Solutions:

(a) The coordinator will not receive either a yes or a no message from the subordinate, and will presume to abort the transaction. It will instruct all the subordinates that replied yes to abort as well, and remove the transaction from the transaction table. Once the subordinate recovers, it will check with the coordinator, which will send the default abort message and the sub transaction will also be aborted.

(b) The situation is the same as above, since the subordinate has not yet made a decision. The coordinator will abort the transaction, after waiting for some time. When the subordinate recovers, it will check with the coordinator, receive an abort message, and will abort the sub transaction.

(c) Since there is no prepare, commit or abort log record, T can be unilaterally aborted and undone, and an end log record written. Since the coordinator did not send the prepare message before it crashed, the subordinates could not have voted to commit.

(d) The subordinates (that voted yes) are waiting to hear from the coordinator the final decision, but fail to do so because the coordinator has failed. These sites are now effectively blocked and will send messages to the coordinator repeatedly. When the coordinator come back up, it finds an abort log record, and will undo T, and remove it from its transaction table. Any contacting subordinate will be told to abort its sub transaction of T.

Question 4:

Suppose there are three sites; A, B and C, all three contain copies of objects O1, O2 and O3. Assume that the read-any write-all technique is used.

There are three transactions, T1, T2 and T3 issued at sites A, B and C accordingly.

T1	T2	T3
read O1		
	read O2	
		read O3
write O2		
	write O3	
		write O1

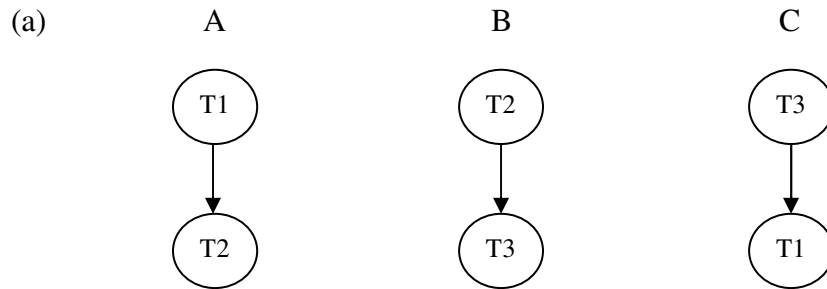
(a) Draw the local waits-for graph at each site.

(b) Does deadlock exist in this situation?

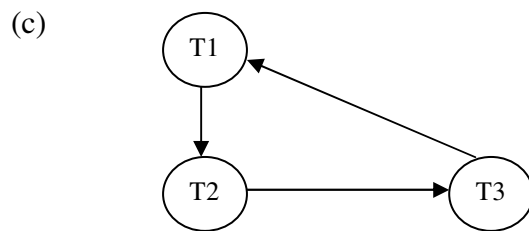
(c) Draw the global waits-for graph.

Solution:

T1 which wants to read O1 and write O2, obtains an S lock on O1 and an X lock on O2 at site A, then requests an X lock on O2 on site B and C. **T2** which wants to read O2 and write O3, meanwhile obtains an S lock on O2 and an X lock on O3 at site B and then requests an X lock on O3 at site A and C. **T3** which wants to read O3 and write O1, meanwhile obtains an S lock on O3 and an X lock on O1 at site C and then requests an X lock on O1 at site A and B.



(b) Yes.



There is a cycle in the global waits-for graph thus there is a deadlock.

T4. System Integration

Question : A publication database is being maintained at two different sites. The tables maintained at the two sites, db1 and db2, are given below. These two databases now have to be integrated into a global conceptual schema.

Schema for db1:

scientist [name, age, state-birth]
publication [title, keyword, year]
book [num, title, year]
series [name, editor]
belongs-to [book-nr, series-name]
publisher-of [series-name, pub-name]
written-by [publ-title, auth-name]
author [name, address]
of-interest [auth-name, topic]

Schema for db2:

scientist [name, position, degree, city-of-birth]
borrower [pub-title, sci-name]
dealt-with [paper-title, topic-code]
paper [title, author]
journal [title, stack]
proceedings [title, stack]
book [title, author, stack]
topic [code, name]
concerns [pub-title, topic-code]
belongs-to-proc [paper-title, proc-title]

Integrate the above by defining views, where each view definition is given by the following statement:

```
CREATE VIEW view-name (attr1, attr2, ... attrn) AS SELECT statement
```

Tips: Try to establish semantic similarity between the attributes that are named and structured differently in the two schemas. For example written-by@db1.publ-title is the same as paper@db2.title

Not all attributes are found in both schemas. For example scientist@db1.age is not found in scientist@db2. Similarly scientist@db2.position is not found in scientist@db1.

SELECT statements used in the view creation would often have unions among result relations retrieved from various database relations belonging to different sites.

For example to create a global view of all topics, we can

```
CREATE VIEW topic (topic-name) AS
( SELECT keyword FROM publication@db1 )
UNION
( SELECT name FROM topic@db2 );
```

Solution:

```
CREATE VIEW paper (title) AS
( SELECT title FROM publication@db1 )
UNION
( SELECT title FROM paper@db2 );
```

```
CREATE VIEW written-by (title, author) AS
( SELECT publ-title, auth-name FROM written-by@db1 )
UNION
( SELECT title, author FROM paper@db2 )
UNION
( SELECT title, author FROM book@db2 );
```

```
CREATE VIEW topic (name) AS
( SELECT keyword FROM publication@db1 )
UNION
( SELECT name FROM topic@db2 );
```

```
CREATE VIEW paper-topic (title, code) AS  
( SELECT paper-title, topic-code FROM dealt-with@db2 );
```

```
CREATE VIEW concerns (pub-name, topic) AS  
( SELECT title, keyword FROM publication@db1 )  
UNION  
( SELECT pub-title, name FROM topic@db2, concerns@db2 WHERE  
code=topic-code );
```

```
CREATE VIEW publication-stack (title, stack, kind) AS  
( SELECT title, stack, 'Journal' FROM journal@db2 )  
UNION  
( SELECT title, stack, 'Proceedings', FROM proceedings@db2 )  
UNION  
( SELECT title, stack, 'Book' FROM book@db2 );
```

```
CREATE VIEW scientist (name, position, degree, age, city-of-birth, state-  
of-birth) AS  
( SELECT s2.name, position, degree, age, city-of-birth, state-of-birth FROM  
scientist@db1 s1, scientist@db2 s2 WHERE s1.name = s2.name )  
UNION  
( SELECT s2.name, position, degree, NULL, city-of-birth, NULL FROM  
scientist@db2 s2 WHERE s2.name NOT IN ( SELECT s1.name FROM  
scientist@db1 s1 ))  
UNION  
( SELECT s1.name, NULL, NULL, age, NULL, state-of-birth FROM  
scientist@db1 s1 WHERE s1.name NOT IN ( SELECT s2.name FROM  
scientist@db2 s2 ));
```

Question 2:

Stockbroker A has a database about daily stock prices and trading volumes, where attribute “company” consists of stock exchange code (such as ECP, TLS, etc):

```
stock(company, date, low, high, close, volume)
```

Stockbroker B also has a database about daily stock closing prices as well as this stockbroker’s trading information (the amount bought, sold and held), where there is one table for each stock (such as ECP, TLS, etc):

```
ECP(day, month, year, price, bought, sold, balance),  
TLS(day, month, year, price, bought, sold, balance)...
```

Primary keys are underlined.

Identify the following data representation differences in the two databases (using one example from the above databases for each problem).

- (a) Format conflicts
- (b) Structural differences
- (c) Missing Data

Solution:

(a) Format conflicts Stock.date and XXX.(day,month,year).

(b) Structural differences Company code(eg, ECP) as values in A, but as table names in B.

(c) Missing data Stock.volume (and many others).

Question 3:

Discuss integration issues in the process of global schema design for the following two existing database systems.

System 1

The following information is stored and maintained. (A sample of data is given.)

Product	(ID#, name, producer, price)
	32 A IBM 500
	39 A MS 475
	141 A HP 512
	34 B IBM 32
	017 C HP 140
	211 D IBM 750

System 2

For every company (producer) they store information in different tables. Additionally, they store the manager (sales rep) responsible for a given product.

The following is a sample of data:

IBM	(ID#,	name,	slsrep,	price)
	32	A	Bob	550
	211	D	Bob	750
	315	E	John	600

HP	(ID#,	name,	slsrep,	price)
	141	A	Keith	580
	018	C	Keith	200
	017	C	Keith	160

MS	(ID#,	name,	slsrep,	price)
	15	A	John	200
	39	A	John	490
	200	E	Bob	300

Discussion:

- System 1 and 2 have different structures of their databases.
- System 1 and 2 have different prices of the same products (identified by ID#).
- System 2 stores an additional attribute, slsrep.

Question 4: Discussion question

Why is query optimisation in Mutidatabase Systems difficult? Compare this to parallel computation.

- Heterogeneity
- Each local system has its own policy for optimisation
- Each local system has its own applications and own strategies
- Unable to influence local queries