

10

Managing Data Concurrency

ORACLE

Objectives

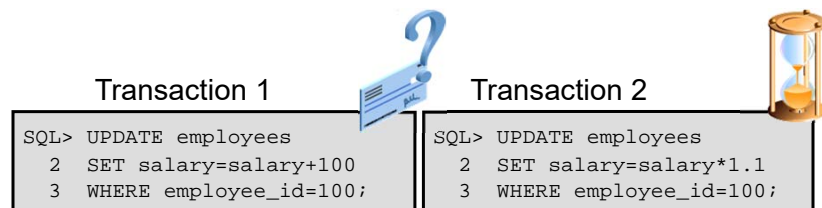
After completing this lesson, you should be able to:

- Describe the locking mechanism and how Oracle manages data concurrency
- Monitor and resolve locking conflicts

ORACLE

Locks

- Prevent multiple sessions from changing the same data at the same time
- Are automatically obtained at the lowest possible level for a given statement
- Do not escalate



ORACLE

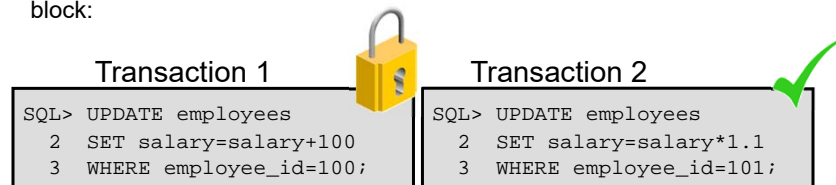
10 - 3

Locking Mechanism

- High level of data concurrency:
 - Row-level locks for inserts, updates, and deletes
 - No locks required for queries
- Automatic queue management
- Locks held until the transaction ends (with a commit or rollback operation)

Example

Assume that the rows for `EMPLOYEE_ID` 100 and 101 reside in the same block:



ORACLE

10 - 4

Data Concurrency

Time: 09:00:00	Transaction 1	UPDATE hr.employees SET salary=salary+100 WHERE employee_id=100;
	Transaction 2	UPDATE hr.employees SET salary=salary+100 WHERE employee_id=101;
	Transaction 3	UPDATE hr.employees SET salary=salary+100 WHERE employee_id=102;

	Transaction x	UPDATE hr.employees SET salary=salary+100 WHERE employee_id=xxx;

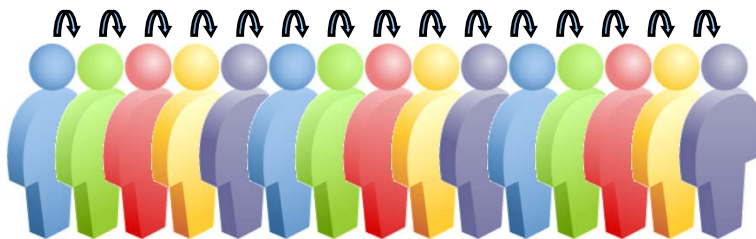
ORACLE

10 - 5

Enqueue Mechanism

The enqueue mechanism keeps track of:


- Sessions waiting for locks
- Requested lock mode
- Order in which sessions requested the lock



ORACLE

10 - 6

Lock Conflicts

Transaction 1	Time	Transaction 2
UPDATE employees SET salary=salary+100 WHERE employee_id=100; 1 row updated.	9:00:00	UPDATE employees SET salary=salary+100 WHERE employee_id=101; 1 row updated.
UPDATE employees SET COMMISSION_PCT=2 WHERE employee_id=101; Session waits enqueued due to lock conflict.	9:00:05 	SELECT sum(salary) FROM employees; SUM(SALARY) ----- 692634
Session still waiting!	16:30:00	Many selects, inserts, updates, and deletes during the last 7.5 hours, but no commits or rollbacks!
1 row updated. Session continues.	16:30:01	commit;

ORACLE

10 - 7

Possible Causes of Lock Conflicts

- Uncommitted changes
- Long-running transactions
- Unnecessarily high locking levels



ORACLE

10 - 8

Resolving Lock Conflicts

To resolve a lock conflict:

- Have the session holding the lock commit or roll back
- Terminate the session holding the lock (in an emergency)

10 - 9

ORACLE

Resolving Lock Conflicts by Using SQL

SQL statements can be used to determine the blocking session and kill it.

1 SQL> SELECT sid, serial#, username
2 FROM v\$session WHERE sid IN
3 (SELECT blocking_session FROM v\$session);

Result:


SID	SERIAL#	USERNAME
144	8982	HR

2 SQL> ALTER SYSTEM KILL SESSION '144,8982' immediate;

10 - 10

ORACLE

Deadlocks



Transaction 1		Transaction 2
UPDATE employees SET salary = salary x 1.1 WHERE employee_id = 1000;	9:00	UPDATE employees SET manager = 1342 WHERE employee_id = 2000;
UPDATE employees SET salary = salary x 1.1 WHERE employee_id = 2000;	9:15	UPDATE employees SET manager = 1342 WHERE employee_id = 1000;
ORA-00060: Deadlock detected while waiting for resource	9:16	

ORACLE

10 - 11

Summary

In this lesson, you should have learned how to:

- Describe the locking mechanism and how Oracle manages data concurrency
- Monitor and resolve locking conflicts

ORACLE

10 - 12