Chapter 9

TRANSACTION MANAGEMENT AND CONCURRENCY CONTROL

In this chapter, you will learn:

- What a database transaction is and what its properties are
- How database transactions are managed
- What concurrency control is and what role it plays in maintaining the database's integrity
- What locking methods are and how they work
- How database recovery management is used to maintain database integrity

What is a Transaction?

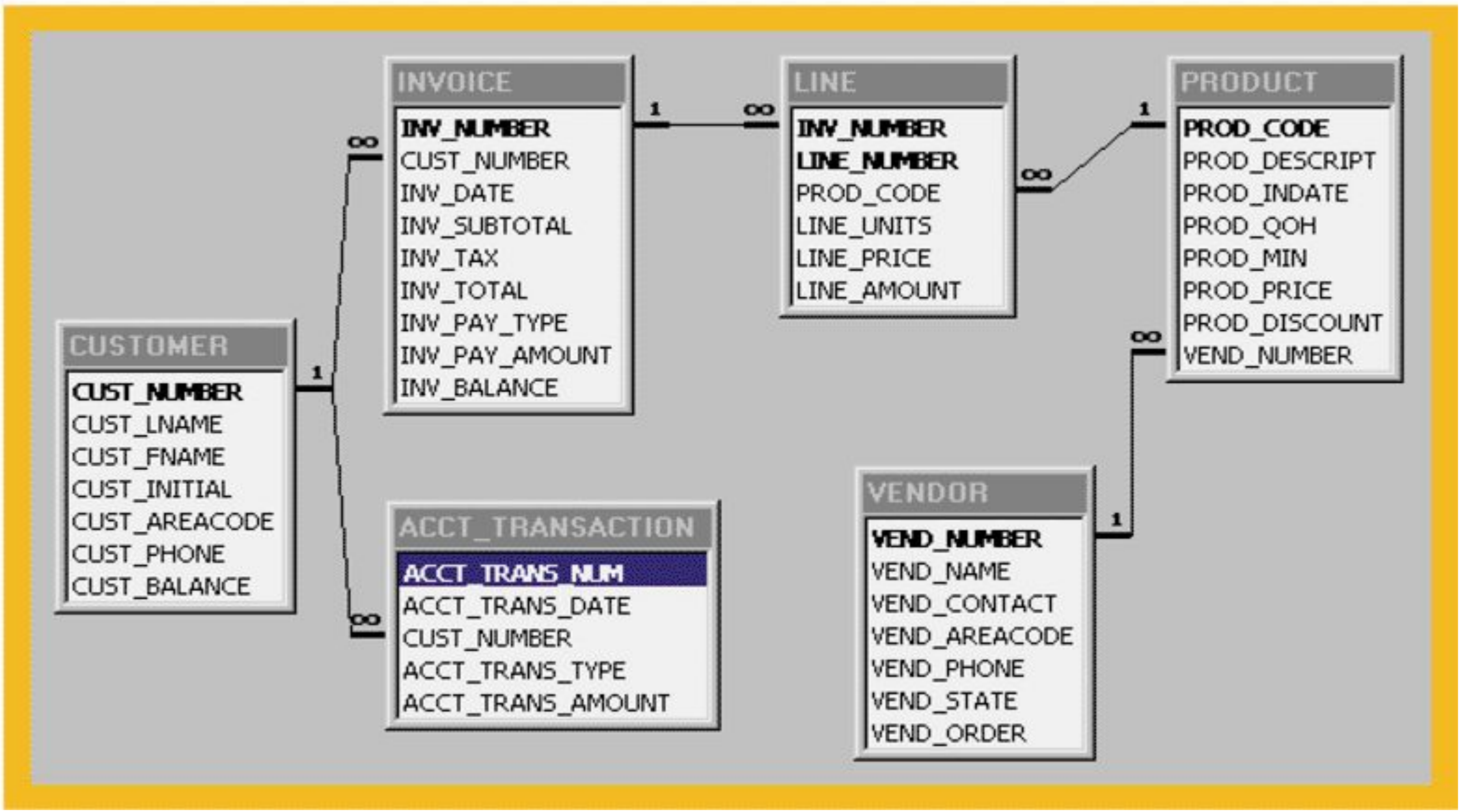
- Any action that reads from and/or writes to a database may consist of
  - Simple SELECT statement to generate a list of table contents
  - A series of related UPDATE statements to change the values of attributes in various tables
  - A series of INSERT statements to add rows to one or more tables
  - A combination of SELECT, UPDATE, and INSERT statements

What is a Transaction? (continued)

- A *logical* unit of work that must be either entirely completed or aborted
- Successful transaction changes the database from one consistent state to another
  - One in which all data integrity constraints are satisfied
- Most real-world database transactions are formed by two or more database requests
  - The equivalent of a single SQL statement in an application program or transaction

The Relational Schema for the Ch09_SaleCo Database

FIGURE 9.1 THE RELATIONAL SCHEMA FOR THE CH09_SALECO DATABASE

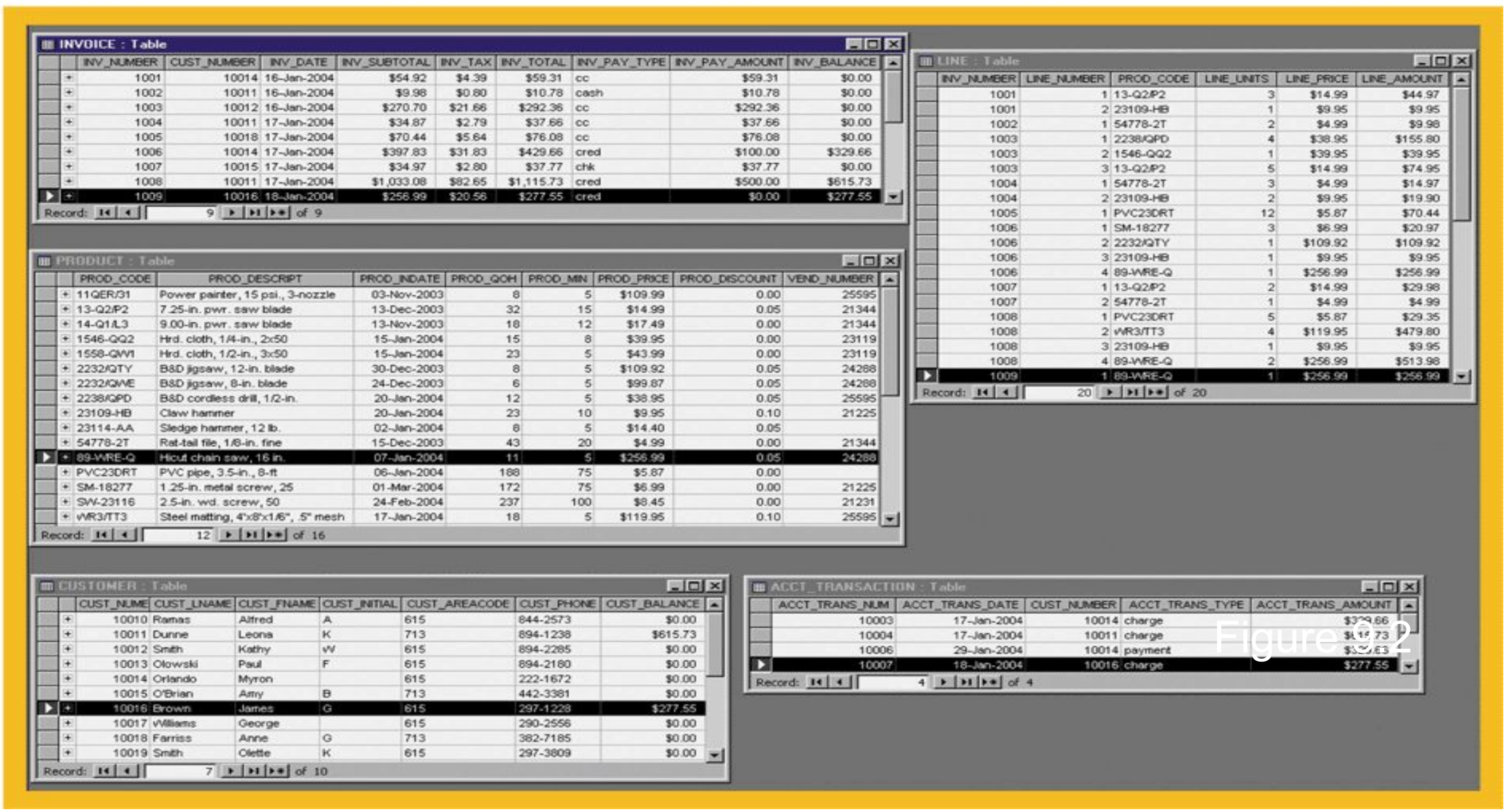


Evaluating Transaction Results

- Not all transactions update the database
- SQL code represents a transaction because database was accessed
- Improper or incomplete transactions can have a devastating effect on database integrity
  - Some DBMSs provide means by which user can define enforceable constraints based on business rules
  - Other integrity rules are enforced automatically by the DBMS when table structures are properly defined, thereby letting the DBMS validate some transactions

Tracing the Transaction in the Ch09_SaleCo Database

FIGURE 9.2 Tracing the Transaction in the Ch09_SaleCo Database



INVOICE : Table

INV_NUMBER	CUST_NUMBER	INV_DATE	INV_SUBTOTAL	INV_TAX	INV_TOTAL	INV_PAY_TYPE	INV_PAY_AMOUNT	INV_BALANCE
1001	10014	16-Jan-2004	$54.92	$4.39	$59.31	cc	$59.31	$0.00
1002	10011	16-Jan-2004	$9.98	$0.80	$10.78	cash	$10.78	$0.00
1003	10012	16-Jan-2004	$270.70	$21.66	$292.36	cc	$292.36	$0.00
1004	10011	17-Jan-2004	$34.87	$2.79	$37.66	cc	$37.66	$0.00
1005	10018	17-Jan-2004	$70.44	$5.64	$76.08	cc	$76.08	$0.00
1006	10014	17-Jan-2004	$397.83	$31.83	$429.66	cred	$100.00	$329.66
1007	10015	17-Jan-2004	$34.97	$2.80	$37.77	chk	$37.77	$0.00
1008	10011	17-Jan-2004	$1,033.08	$82.65	$1,115.73	cred	$500.00	$615.73
1009	10016	18-Jan-2004	$256.99	$20.56	$277.55	cred	$0.00	$277.55

Record: 9 of 9

LINE : Table

INV_NUMBER	LINE_NUMBER	PROD_CODE	LINE_UNITS	LINE_PRICE	LINE_AMOUNT
1001	1	13-Q2/P2	3	$14.99	$44.97
1001	2	23109-HB	1	$9.95	$9.95
1002	1	54778-2T	2	$4.99	$9.98
1003	1	2238/QPD	4	$38.95	$155.80
1003	2	1546-QQ2	1	$39.95	$39.95
1003	3	13-Q2/P2	5	$14.99	$74.95
1004	1	54778-2T	3	$4.99	$14.97
1004	2	23109-HB	2	$9.95	$19.90
1005	1	PVC23DRT	12	$5.87	$70.44
1006	1	SM-18277	3	$6.99	$20.97
1006	2	2232/QTY	1	$109.92	$109.92
1006	3	23109-HB	1	$9.95	$9.95
1006	4	89-WRE-Q	1	$256.99	$256.99
1007	1	13-Q2/P2	2	$14.99	$29.98
1007	2	54778-2T	1	$4.99	$4.99
1008	1	PVC23DRT	5	$5.87	$29.35
1008	2	WR3/TT3	4	$119.95	$479.80
1008	3	23109-HB	1	$9.95	$9.95
1008	4	89-WRE-Q	2	$256.99	$513.98
1009	1	89-WRE-Q	1	$256.99	$256.99

Record: 20 of 20

PRODUCT : Table

PROD_CODE	PROD_DESCRIPT	PROD_INDATE	PROD_QOH	PROD_MIN	PROD_PRICE	PROD_DISCOUNT	VEND_NUMBER
11QER/31	Power painter, 15 psi., 3-nozzle	03-Nov-2003	8	5	$109.99	0.00	25595
13-Q2/P2	7.25-in. pwr. saw blade	13-Dec-2003	32	15	$14.99	0.05	21344
14-Q1/L3	9.00-in. pwr. saw blade	13-Nov-2003	18	12	$17.49	0.00	21344
1546-QQ2	Hrd. cloth, 1/4-in., 2x50	15-Jan-2004	15	8	$39.95	0.00	23119
1558-QW1	Hrd. cloth, 1/2-in., 3x50	15-Jan-2004	23	5	$43.99	0.00	23119
2232/QTY	B&D jigsaw, 12-in. blade	30-Dec-2003	8	5	$109.92	0.05	24288
2232/QWE	B&D jigsaw, 8-in. blade	24-Dec-2003	6	5	$99.87	0.05	24288
2238/QPD	B&D cordless drill, 1/2-in.	20-Jan-2004	12	5	$38.95	0.05	25595
23109-HB	Claw hammer	20-Jan-2004	23	10	$9.95	0.10	21225
23114-AA	Sledge hammer, 12 lb.	02-Jan-2004	8	5	$14.40	0.05	
54778-2T	Rat-tail file, 1/8-in. fine	15-Dec-2003	43	20	$4.99	0.00	21344
89-WRE-Q	Hicut chain saw, 16 in.	07-Jan-2004	11	5	$256.99	0.05	24288
PVC23DRT	PVC pipe, 3.5-in., 8-ft	06-Jan-2004	188	75	$5.87	0.00	
SM-18277	1.25-in. metal screw, 25	01-Mar-2004	172	75	$6.99	0.00	21225
SW-23116	2.5-in. wd. screw, 50	24-Feb-2004	237	100	$8.45	0.00	21231
WR3/TT3	Steel matting, 4'x8'x1/6", .5" mesh	17-Jan-2004	18	5	$119.95	0.10	25595

Record: 12 of 16

CUSTOMER : Table

CUST_NUMBER	CUST_LNAME	CUST_FNAME	CUST_INITIAL	CUST_AREACODE	CUST_PHONE	CUST_BALANCE
10010	Ramas	Alfred	A	615	844-2573	$0.00
10011	Dunne	Leona	K	713	894-1238	$615.73
10012	Smith	Kathy	W	615	894-2285	$0.00
10013	Olowski	Paul	F	615	894-2180	$0.00
10014	Orlando	Myron		615	222-1672	$0.00
10015	O'Brian	Amy	B	713	442-3381	$0.00
10016	Brown	James	G	615	297-1228	$277.55
10017	Williams	George		615	290-2556	$0.00
10018	Farriss	Anne	G	713	382-7185	$0.00
10019	Smith	Olette	K	615	297-3809	$0.00

Record: 7 of 10

ACCT_TRANSACTION : Table

ACCT_TRANS_NUM	ACCT_TRANS_DATE	CUST_NUMBER	ACCT_TRANS_TYPE	ACCT_TRANS_AMOUNT
10003	17-Jan-2004	10014	charge	$3[illegible]9.66
10004	17-Jan-2004	10011	charge	$6[illegible].73
10006	29-Jan-2004	10014	payment	[illegible]
10007	18-Jan-2004	10016	charge	$277.55

Record: 4 of 4

Figure 9.2

Transaction Properties

- Atomicity
  - Requires that *all* operations (SQL requests) of a transaction be completed

Transaction-Level Atomicity: The entire goal of a transaction, a set of SQL statements executed together as a unit of work, is to take the database from one consistent state to another consistent state. To accomplish this goal, transactions are atomic as well—the entire set of successful work performed by a transaction is either entirely committed and made permanent or rolled back and undone. Just like a statement, the transaction is an atomic unit of work. Upon receipt of “success” from the database after committing a transaction, you know that all of the work performed by the transaction has been made persistent.

- Durability
  - Indicates permanence of database’s consistent state

Transaction Properties (continued)

- Serializability
  - Ensures that the concurrent execution of several transactions yields consistent results
- Isolation
  - Data used during execution of a transaction cannot be used by second transaction until first one is completed

Integrity Constraints and Transactions

Integrity Constraints and Transactions It is interesting to note exactly when integrity constraints are checked. By default, integrity constraints are checked after the entire SQL statement has been processed. There are also deferrable constraints that permit the validation of integrity constraints to be postponed until either the application requests they be validated by issuing a SET CONSTRAINTS ALL IMMEDIATE command or upon issuing a COMMIT.

IMMEDIATE Constraints For the first part of this discussion, we'll assume that constraints are in IMMEDIATE mode, which is the norm. In this case, the integrity constraints are checked immediately after the entire SQL statement has been processed. Note that I used the term "SQL statement," not just "statement." If I have many SQL statements in a PL/SQL stored procedure, each SQL statement will have its integrity constraints validated immediately after its individual execution, not after the stored procedure completes. So, why are constraints validated after the SQL statement executes? Why not during? This is because it is very natural for a single statement to make individual rows in a table momentarily inconsistent. Taking a look at the partial work by a statement would result in Oracle rejecting the results, even if the end result would be OK. For example, suppose we have a table like this:

```
EODA@ORA12CR1> create table t ( x int unique );
Table created.

EODA@ORA12CR1> insert into t values ( 1 );
1 row created.
EODA@ORA12CR1> insert into t values ( 2 );
1 row created.

EODA@ORA12CR1> commit;
Commit complete.
```

And we want to execute a multiple-row UPDATE:

```
EODA@ORA12CR1> update t set x=x-1;
2 rows updated.
```

Transaction Management with SQL

- ANSI has defined standards that govern SQL database transactions
- Transaction support is provided by two SQL statements: COMMIT and ROLLBACK
- ANSI standards require that, when a transaction sequence is initiated by a user or an application program,
  - it must continue through all succeeding SQL statements until one of four events occurs

The Transaction Log

- Stores
  - A record for the beginning of transaction
  - For each transaction component (SQL statement)
    - Type of operation being performed (update, delete, insert)
    - Names of objects affected by the transaction (the name of the table)
    - “Before” and “after” values for updated fields
    - Pointers to previous and next transaction log entries for the same transaction
  - The ending (COMMIT) of the transaction

A Transaction Log

TABLE 9.1 A Transaction Log

TRL ID	TRX NUM	PREV PTR	NEXT PTR	OPERATION	TABLE	ROW ID	ATTRIBUTE	BEFORE VALUE	AFTER VALUE
341	101	Null	352	START	****Start Transaction				
352	101	341	363	UPDATE	PRODUCT	1558-QW1	PROD_QOH	25	23
363	101	352	365	UPDATE	CUSTOMER	10011	CUST_BALANCE	525.75	615.73
365	101	363	Null	COMMIT	**** End of Transaction				

TRL_ID = Transaction log record ID

PTR = Pointer to a transaction log record ID

TRX_NUM = Transaction number

(Note: The transaction number is automatically assigned by the DBMS.)

Concurrency Control

- Coordination of simultaneous transaction execution in a multiprocessing database system
- Objective is to ensure transaction serializability in a multiuser database environment

Concurrency Control

- Important simultaneous execution of transactions over a shared database can create several data integrity and consistency problems
  - lost updates
  - uncommitted data
  - inconsistent retrievals

Normal Execution of Two Transactions

TABLE 9.2 NORMAL EXECUTION OF TWO TRANSACTIONS

TIME	TRANSACTION	STEP	STORED VALUE
1	T1	Read PROD_QOH	35
2	T1	PROD_QOH = 35 + 100	
3	T1	Write PROD_QOH	135
4	T2	Read PROD_QOH	135
5	T2	PROD_QOH = 135 – 30	
6	T2	Write PROD_QOH	105

Lost Updates

TABLE 9.3 LOST UPDATES

TIME	TRANSACTION	STEP	STORED VALUE
1	T1	Read PROD_QOH	35
2	T2	Read PROD_QOH	35
3	T1	PROD_QOH = 35 + 100	
4	T2	PROD_QOH = 35 – 30	
5	T1	Write PROD_QOH **(Lost update)**	135
6	T2	Write PROD_QOH	5

Correct Execution of Two Transactions

TABLE 9.4 Correct Execution of Two Transactions

TIME	TRANSACTION	STEP	STORED VALUE
1	T1	Read PROD_QOH	35
2	T1	PROD_QOH = 35 + 100	
3	T1	Write PROD_QOH	135
4	T1	*****ROLLBACK *****	35
5	T2	Read PROD_QOH	35
6	T2	PROD_QOH = 35 – 30	
7	T2	Write PROD_QOH	5

An Uncommitted Data Problem

TABLE 9.5 An Uncommitted Data Problem

TIME	TRANSACTION	STEP	STORED VALUE
1	T1	Read PROD_QOH	35
2	T1	PROD_QOH = 35 + 100	
3	T1	Write PROD_QOH	135
4	T2	Read PROD_QOH **(Read uncommitted data)**	135
5	T2	PROD_QOH = 135 – 30	
6	T1	***** **ROLLBACK** *****	35
7	T2	Write PROD_QOH	105

Retrieval During Update

TABLE 9.6 Retrieval During Update

TRANSACTION T1	TRANSACTION T2
SELECT SUM(PROD_QOH) FROM PRODUCT	UPDATE PRODUCT SET PROD_QOH = PROD_QOH + 10 WHERE PROD_CODE = '1546-QQ2'
	UPDATE PRODUCT SET PROD_QOH = PROD_QOH – 10 WHERE PROD_CODE = '1558-QW1'
	COMMIT;

Transaction Results: Data Entry Correction

TABLE 9.7 TRANSACTION RESULTS: DATA ENTRY CORRECTION

	BEFORE	AFTER
PROD_CODE	PROD_QOH	PROD_QOH
11QER/31	8	8
13-Q2/P2	32	32
1546-QQ2	15	(15 + 10) → 25
1558-QW1	23	(23 – 10) → 13
2232-QTY	8	8
2232-QWE	6	6
Total	**92**	**92**

Inconsistent Retrievals

TABLE 9.8 INCONSISTENT RETRIEVALS

TIME	TRANSACTION	ACTION	VALUE	TOTAL
1	T1	Read PROD_QOH for PROD_CODE = '11QER/31'	8	8
2	T1	Read PROD_QOH for PROD_CODE = '13-Q2/P2'	32	40
3	T2	Read PROD_QOH for PROD_CODE = '1546-QQ2'	15	
4	T2	PROD_QOH = 15 + 10		
5	T2	Write PROD_QOH for PROD_CODE = '1546-QQ2'	25	
6	T1	Read PROD_QOH for PROD_CODE = '1546-QQ2'	25	(After) 65
7	T1	Read PROD_QOH for PROD_CODE = '1158-QW1'	23	(Before) 88
8	T2	Read PROD_QOH for PROD_CODE = '1558-QW1'	23	
9	T2	PROD_QOH = 23 – 10		
10	T2	Write PROD_QOH for PROD_CODE = '1558-QW1'	13	
11	T2	***** COMMIT *****		
12	T1	Read PROD_QOH for PROD_CODE = '2232-QTY'	8	96
13	T1	Read PROD_QOH for PROD_CODE = '2232-QWE'	6	102

The Scheduler

- Special DBMS program: establishes order of operations within which concurrent transactions are executed
- Interleaves the execution of database operations to ensure serializability and isolation of transactions

The Scheduler (continued)

- Bases its actions on concurrency control algorithms
- Ensures computer's central processing unit (CPU) is used efficiently
- Facilitates data isolation to ensure that two transactions do not update the same data element at the same time

Queries ?