**44-560 Advanced Topics in Database Systems**

**Transaction Management KEY**

1. Problem 1 (parts a – d only), page 464, at the end of Chapter 10.
   1. How many database requests can you identify for an inventory update for both PRODUCT and PART?

Depending in how the SQL statements are written, there are two correct answers: 4 or 2.

* 1. Using SQL, write each database request you identified in problem 1.

The database requests are shown in the following table.

|  |  |
| --- | --- |
| **Four SQL statements** | **Two SQL statements** |
| UPDATE PRODUCT  SET PROD\_QOH = PROD\_OQH + 1  WHERE PROD\_CODE = ‘ABC’  UPDATE PART  SET PART\_QOH = PART\_OQH - 1  WHERE PART\_CODE = ‘A’  UPDATE PART  SET PART\_QOH = PART\_OQH - 1  WHERE PART\_CODE = ‘B’  UPDATE PART  SET PART\_QOH = PART\_OQH - 1  WHERE PART\_CODE = ‘C’ | UPDATE PRODUCT  SET PROD\_QOH = PROD\_OQH + 1  WHERE PROD\_CODE = ‘ABC’  UPDATE PART  SET PART\_QOH = PART\_OQH - 1  WHERE PART\_CODE = ‘A’ OR  PART\_CODE = ‘B’ OR  PART\_CODE = ‘C’ |

* 1. Write the complete transaction(s).

The transactions are shown in the following table.

|  |  |
| --- | --- |
| **Four SQL statements** | **Two SQL statements** |
| BEGIN TRANSACTION  UPDATE PRODUCT  SET PROD\_QOH = PROD\_OQH + 1  WHERE PROD\_CODE = ‘ABC’  UPDATE PART  SET PART\_QOH = PART\_OQH - 1  WHERE PART\_CODE = ‘A’  UPDATE PART  SET PART\_QOH = PART\_OQH - 1  WHERE PART\_CODE = ‘B’  UPDATE PART  SET PART\_QOH = PART\_OQH - 1  WHERE PART\_CODE = ‘C’  COMMIT; | BEGIN TRANSACTION  UPDATE PRODUCT  SET PROD\_QOH = PROD\_OQH + 1  WHERE PROD\_CODE = ‘ABC’  UPDATE PART  SET PART\_QOH = PART\_QOH - 1  WHERE PART\_CODE = ‘A’ OR  PART\_CODE = ‘B’ OR  PART\_CODE = ‘C’  COMMIT; |

* 1. Write the transaction log, using Table 10.1 as your template.

Note that the current values for quanity-on-hand of product ABC and parts A, B, and C are given in Table P10.1 of your textbook.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **TRL**  **ID** | **TRX**  **NUM** | **PREV**  **PTR** | **NEXT**  **PTR** | **OPERATION** | **TABLE** | **ROW**  **ID** | **ATTRIBUTE** | **BEFORE**  **VALUE** | **AFTER**  **VALUE** |
| 1 | 1A3 | NULL | 2 | START | \*\*START TRANSACTION |  |  |  |  |
| 2 | 1A3 | 1 | 3 | UPDATE | PRODUCT | ‘ABC’ | PROD\_QOH | 1,205 | 1,206 |
| 3 | 1A3 | 2 | 4 | UPDATE | PART | ‘A’ | PART\_QOH | 567 | 566 |
| 4 | 1A3 | 3 | 5 | UPDATE | PART | ‘B’ | PART\_QOH | 98 | 97 |
| 5 | 1A3 | 4 | 6 | UPDATE | PART | ‘C’ | PART\_QOH | 549 | 548 |
| 6 | 1A3 | 5 | NULL | COMMIT | \*\* END  TRANSACTION |  |  |  |  |

1. Problem 6, pages 464-465, at the end of Chapter 10.
   1. On May 11, 2012, customer ‘10010’ makes a credit purchase (30 days) of one unit of product ‘11QER/31’ with a unit price of $110.00; the tax rate is 8 percent. The invoice number is 10983, and this invoice has only one product line.

**BEGIN TRANSACTION;**

**INSERT INTO INVOICE**

**VALUES (10983, ‘10010’, ‘11-May-2012’, 118.80, ‘30’, ‘OPEN’);**

**INSERT INTO LINE**

**VALUES (10983, 1, ‘11QER/31’, 1, 110.00);**

**UPDATE PRODUCT**

**SET P\_QTYOH = P\_QTYOH – 1**

**WHERE P\_CODE = ‘11QER/31’;**

**UPDATE CUSTOMER**

**SET CUS\_DATELSTPUR = ‘11-May-2012’,**

**CUS\_BALANCE = CUS\_BALANCE + 118.80**

**WHERE CUS\_CODE = ‘10010’;**

**COMMIT;**

* 1. On June 3, 2012, customer ‘10010’ makes a payment of $100 in cash. The payment ID is 3428.

**BEGIN TRANSACTION;**

**INSERT INTO PAYMENTS**

**VALUES (3428, ‘03-Jun-2012’, ‘10010’, 100.00, ‘CASH’, 'None');**

**UPDATE CUSTOMER**

**SET CUS\_DATELSTPMT = ‘03-Jun-2012’,**

**CUS\_BALANCE = CUS\_BALANCE - 100.00**

**WHERE CUS\_CODE = ‘10010’;**

**COMMIT;**

1. Problem 7, page 465, at end of Chapter 10: Create a simple transaction log (using the format shown in Table 10.1) to represent the actions of the two previous transactions.

The transaction log is shown in Table P10 below.

**Table P10.6 The ABC Markets Transaction Log**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **TRL**  **ID** | **TRX**  **NUM** | **PREV**  **PTR** | **NEXT**  **PTR** | **OPERATION** | **TABLE** | **ROW ID** | **ATTRIBUTE** | **BEFORE**  **VALUE** | **AFTER**  **VALUE** |
| 987 | 101 | Null | 1023 | START | \* Start Trx. |  |  |  |  |
| 1023 | 101 | 987 | 1026 | INSERT | INVOICE | 10983 |  |  | 10983, 10010,  11-May-2012, 118.80, 30, OPEN |
| 1026 | 101 | 1023 | 1029 | INSERT | LINE | 10983, 1 |  |  | 10983, 1, 11QER/31, 1, 110.00 |
| 1029 | 101 | 1026 | 1031 | UPDATE | PRODUCT | 11QER/31 | P\_QTYOH | 47 | 46 |
| 1031 | 101 | 1029 | 1032 | UPDATE | CUSTOMER | 10010 | CUS\_BALANCE | 345.67 | 464.47 |
| 1032 | 101 | 1031 | 1034 | UPDATE | CUSTOMER | 10010 | CUS\_DATELSTPUR | 5-May-2006 | 11-May-2012 |
| 1034 | 101 | 1032 | Null | COMMIT | \* End Trx. \* |  |  |  |  |
| 1089 | 102 | Null | 1091 | START | \* Start Trx. |  |  |  |  |
| 1091 | 102 | 1089 | 1095 | INSERT | PAYMENT | 3428 |  |  | 3428, 3-Jun-2012, 10010, 100.00, CASH, None |
| 1095 | 102 | 1091 | 1096 | UPDATE | CUSTOMER | 10010 | CUS\_BALANCE | 464.47 | 364.47 |
| 1096 | 102 | 1095 | 1097 | UPDATE | CUSTOMER | 10010 | CUS\_DATELSTPMT | 2-May-2006 | 3-Jun-2012 |
| 1097 | 102 | 1096 | Null | COMMIT | \* End Trx. |  |  |  |  |

Note: Because we have not shown the table contents, the "before" values in the transaction can be assumed. The "after" value must be computed using the assumed "before" value, plus or minus the transaction value. Note that we combine the "after" values for the INSERT statements into a single cell. There is no “before” value for an insert statement.