```sql
-- 01 update triggers
-- test.db

CREATE TABLE widgetCustomer
(
    id INTEGER PRIMARY KEY,
    name TEXT,
    last_order_id INT
);
CREATE TABLE widgetSale
(
    id INTEGER PRIMARY KEY,
    item_id INT,
    customer_id INT,
    quan INT,
    price INT
);

INSERT INTO widgetCustomer
    (name)
VALUES
    ('Bob');
INSERT INTO widgetCustomer
    (name)
VALUES
    ('Sally');
INSERT INTO widgetCustomer
    (name)
VALUES
    ('Fred');

SELECT *
FROM widgetCustomer;

CREATE TRIGGER newWidgetSale AFTER
INSERT ON
widgetSale
BEGIN
    UPDATE widgetCustomer SET last_order_id = NEW.id WHERE widgetCustomer.id = NEW.customer_id;
END
;

INSERT INTO widgetSale
    (item_id, customer_id, quan, price)
VALUES
    (1, 3, 5, 1995);
INSERT INTO widgetSale
    (item_id, customer_id, quan, price)
VALUES
    (2, 2, 3, 1495);
INSERT INTO widgetSale
    (item_id, customer_id, quan, price)
VALUES
    (3, 1, 1, 2995);
SELECT *
FROM widgetSale;
SELECT *
FROM widgetCustomer;

-- 02 preventing updates
-- test.db

DROP TABLE IF EXISTS widgetSale;

CREATE TABLE widgetSale
(
    id integer primary key,
```

```
    item_id INT,
    customer_id INTEGER,
    quan INT,
    price INT,
    reconciled INT
);
INSERT INTO widgetSale
    (item_id, customer_id, quan, price, reconciled)
VALUES
    (1, 3, 5, 1995, 0);
INSERT INTO widgetSale
    (item_id, customer_id, quan, price, reconciled)
VALUES
    (2, 2, 3, 1495, 1);
INSERT INTO widgetSale
    (item_id, customer_id, quan, price, reconciled)
VALUES
    (3, 1, 1, 2995, 0);
SELECT *
FROM widgetSale;

CREATE TRIGGER updateWidgetSale BEFORE
UPDATE ON widgetSale
BEGIN
    SELECT RAISE(
    ROLLBACK
    , 'cannot update table "widgetSale"') FROM widgetSale
           WHERE id = NEW.id AND reconciled = 1;
END
;

BEGIN TRANSACTION;
UPDATE widgetSale SET quan = 9 WHERE id = 2;
END TRANSACTION;

SELECT *
FROM widgetSale;

-- 03 timestamps
-- test.db

DROP TABLE IF EXISTS widgetSale;
DROP TABLE IF EXISTS widgetCustomer;

CREATE TABLE widgetCustomer
(
    id integer primary key,
    name TEXT,
    last_order_id INT,
    stamp TEXT
);
CREATE TABLE widgetSale
(
    id integer primary key,
    item_id INT,
    customer_id INTEGER,
    quan INT,
    price INT,
    stamp TEXT
);
CREATE TABLE widgetLog
(
    id integer primary key,
    stamp TEXT,
    event TEXT,
    username TEXT,
    tablename TEXT,
```

```
    table_id INT
);

INSERT INTO widgetCustomer
    (name)
VALUES
    ('Bob');
INSERT INTO widgetCustomer
    (name)
VALUES
    ('Sally');
INSERT INTO widgetCustomer
    (name)
VALUES
    ('Fred');
SELECT *
FROM widgetCustomer;

CREATE TRIGGER stampSale AFTER
INSERT ON
widgetSale
BEGIN
    UPDATE widgetSale SET stamp = DATETIME('now') WHERE id = NEW.id;
    UPDATE widgetCustomer SET last_order_id = NEW.id, stamp = DATETIME('now')
            WHERE widgetCustomer.id = NEW.customer_id;
    INSERT INTO widgetLog
        (stamp, event, username, tablename, table_id)
    VALUES
        (DATETIME('now'), 'INSERT', 'TRIGGER', 'widgetSale', NEW.id);
END
;

INSERT INTO widgetSale
    (item_id, customer_id, quan, price)
VALUES
    (1, 3, 5, 1995);
INSERT INTO widgetSale
    (item_id, customer_id, quan, price)
VALUES
    (2, 2, 3, 1495);
INSERT INTO widgetSale
    (item_id, customer_id, quan, price)
VALUES
    (3, 1, 1, 2995);

SELECT *
FROM widgetSale;
SELECT *
FROM widgetCustomer;
SELECT *
FROM widgetLog;

-- restore database
DROP TRIGGER IF EXISTS newWidgetSale;
DROP TRIGGER IF EXISTS updateWidgetSale;
DROP TRIGGER IF EXISTS stampSale;

DROP TABLE IF EXISTS widgetCustomer;
DROP TABLE IF EXISTS widgetSale;
DROP TABLE IF EXISTS widgetLog;
```