CSCI235: Database Systems

Assignment 2

Nicholas Monteleone 5055076

Task 3

1)

Transaction 1	Transaction 2
SELECT NVL(MAX(unit_price), 0) INTO max_unit_price FROM PRODUCT;	
UPDATE PRODUCT SET unit_price = unit_price + 0.01*max_unit_price WHERE units_in_stock > 60;	
	UPDATE PRODUCT SET units_in_stock = units_in_stock – 60 WHERE units_in_stock > 60;
	COMMIT;
UPDATE PRODUCT SET unit_price = unit_price + 0.02*max_unit_price WHERE units_in_stock <= 60;	
COMMIT;	

Some product prices are increased two times because the units_in_stock value is decreased after the price is raised, and then raised again due to the low stock level.

2)

```
SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
DECLARE
        max_unit_price NUMBER(9);
        current_unit_stock NUMBER(9);
BEGIN
        SELECT NVL(MAX(unit_price), 0)
        INTO max_unit_price
        FROM PRODUCT;

        SELECT units_in_stock
        INTO current_unit_stock
        FROM PRODUCT;

        UPDATE PRODUCT
        SET unit_price = unit_price + 0.01*max_unit_price
        WHERE current_unit_stock > 60;

        UPDATE PRODUCT
        SET unit_price = unit_price + 0.02*max_unit_price
        WHERE current_unit_stock <= 60;
COMMIT;
END;
/
```

Transaction 1	Transaction 2
SELECT NVL(MAX(unit_price), 0) INTO max_unit_price FROM PRODUCT;	
SELECT units_in_stock INTO current_unit_stock FROM PRODUCT;	
UPDATE PRODUCT SET unit_price = unit_price + 0.01*max_unit_price WHERE current_unit_stock > 60;	
	UPDATE PRODUCT SET units_in_stock = units_in_stock – 60 WHERE units_in_stock > 60;
	COMMIT;
UPDATE PRODUCT SET unit_price = unit_price + 0.02*max_unit_price WHERE current_unit_stock <= 60;	
COMMIT;	

Value is read into a temporary value one time, so that each statement is consistently updated regardless of external values. Schedule is now conflict serializable, meaning no issues in transaction order will occur.