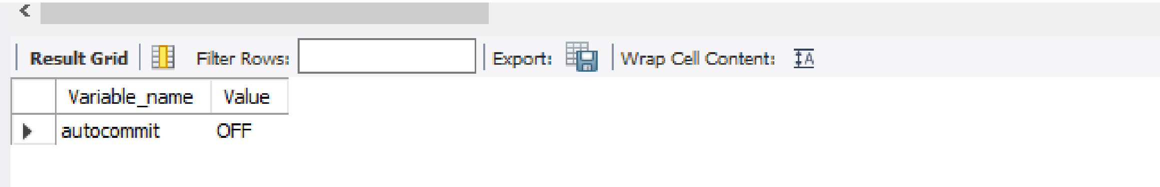
Fundamentals of Data Management

2019HS2 | 101624964 | Jimmy Trac

Credit Task 9.2.2

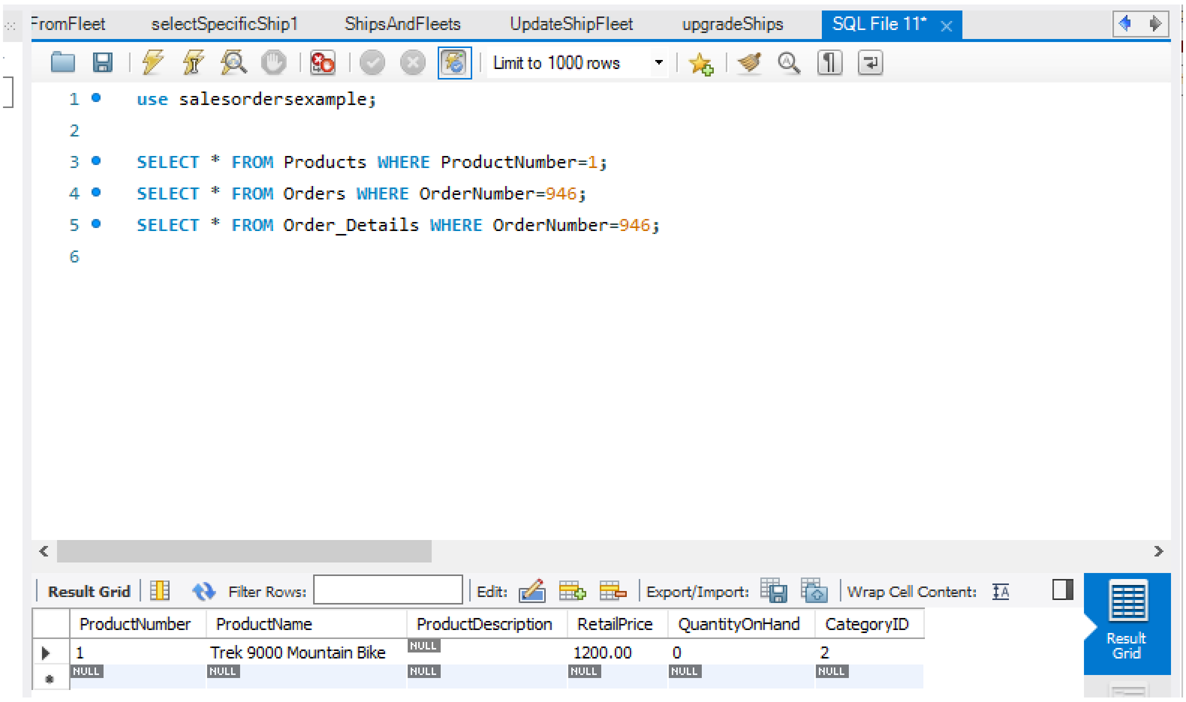
First making sure that `autocommit` is off:



Variable_name	Value
autocommit	OFF

a)

> *Run the first statement of T2 again in your right MySQL Workbench instance. Run all statements of T1 in your left Workbench instance. What do you see?*

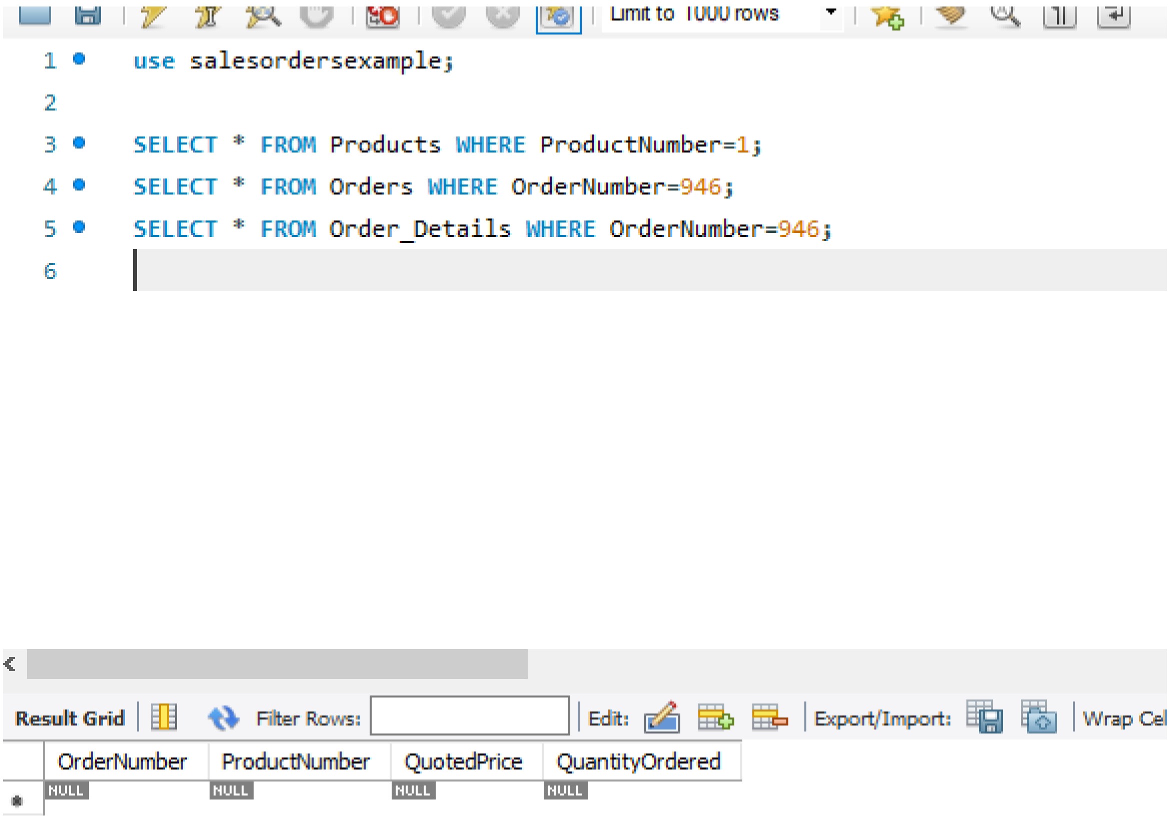


```sql
use salesordersexample;

SELECT * FROM Products WHERE ProductNumber=1;
SELECT * FROM Orders WHERE OrderNumber=946;
SELECT * FROM Order_Details WHERE OrderNumber=946;

```

ProductNumber	ProductName	ProductDescription	RetailPrice	QuantityOnHand	CategoryID
1	Trek 9000 Mountain Bike	NULL	1200.00	0	2
NULL	NULL	NULL	NULL	NULL	NULL

As there have been no commits, there have been no changes (aside from last task 9.2.1)

> *Run the rest of T2 in the right MySQL Workbench. Check again what you can see in your left Workbench.*

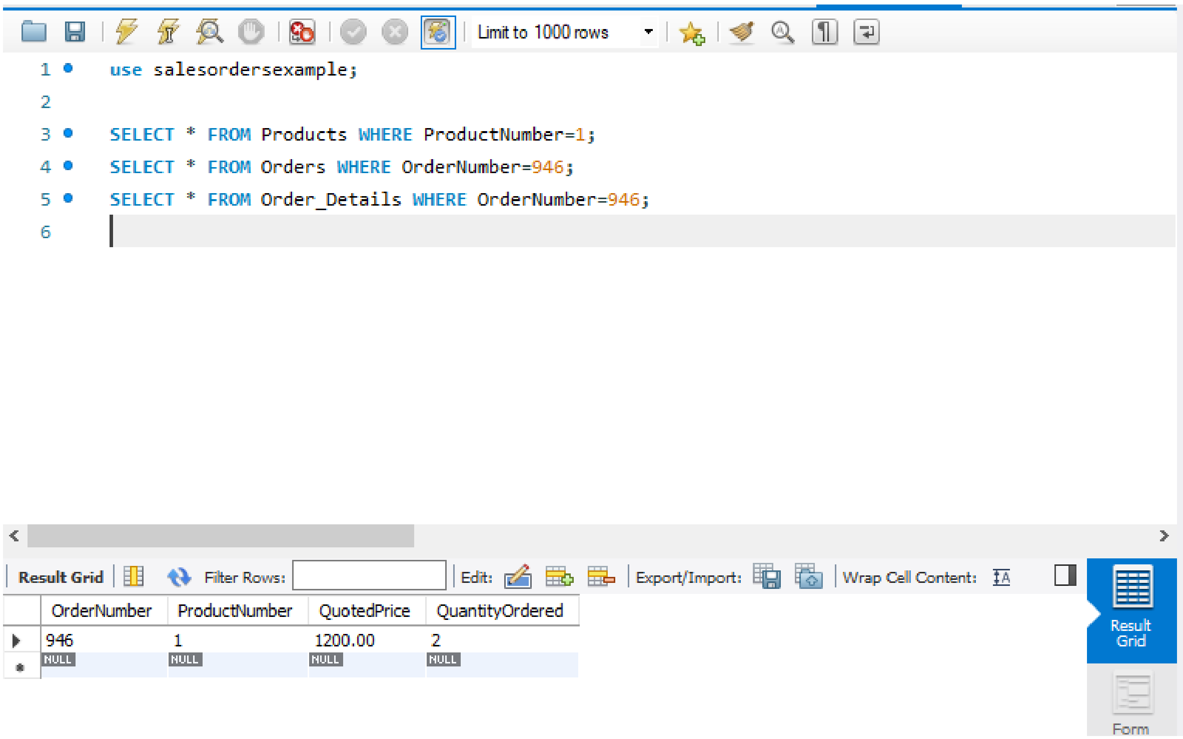


```
use salesordersexample;

SELECT * FROM Products WHERE ProductNumber=1;
SELECT * FROM Orders WHERE OrderNumber=946;
SELECT * FROM Order_Details WHERE OrderNumber=946;
```

OrderNumber	ProductNumber	QuotedPrice	QuantityOrdered
NULL	NULL	NULL	NULL

Again, no commits, no changes.

> *Commit T2 in your right Workbench. Re-run T1 in your left instance. What do you see?*

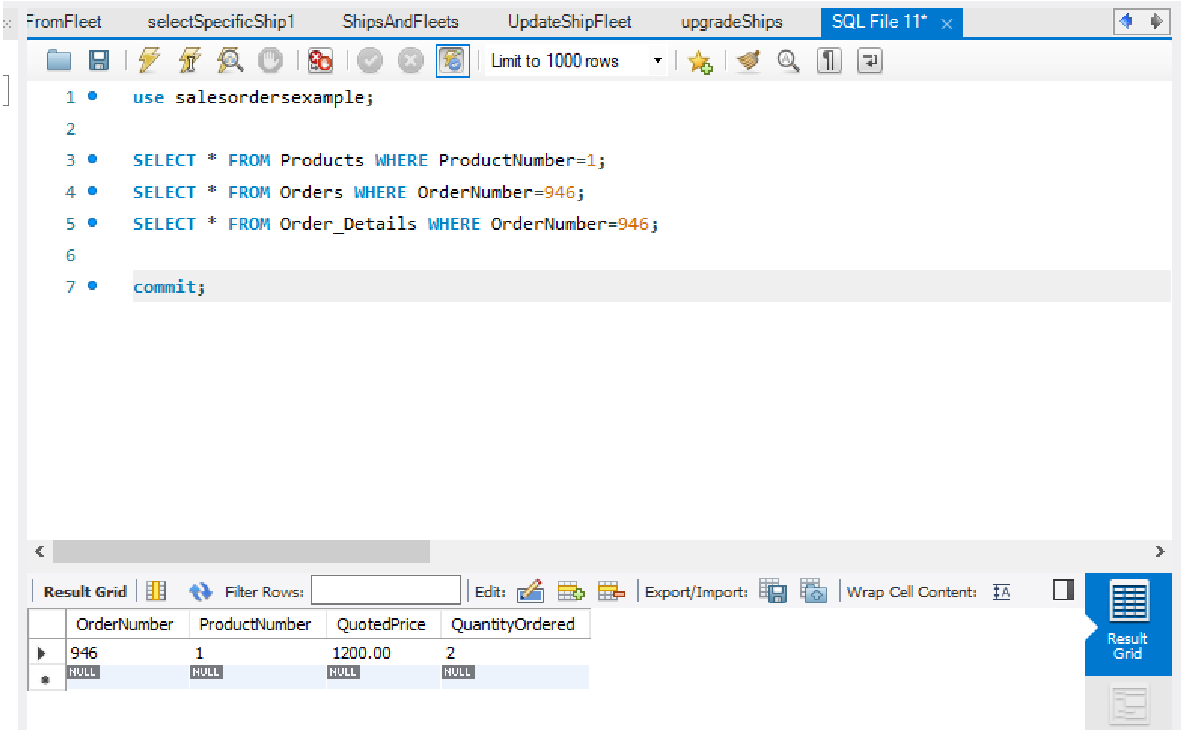


```
use salesordersexample;

SELECT * FROM Products WHERE ProductNumber=1;
SELECT * FROM Orders WHERE OrderNumber=946;
SELECT * FROM Order_Details WHERE OrderNumber=946;
```

OrderNumber	ProductNumber	QuotedPrice	QuantityOrdered
946	1	1200.00	2
NULL	NULL	NULL	NULL

Due to the `READ-COMMIT` isolation level, this time, we can see the commits made by T2

> *Commit T1 in your left Workbench. Re-run T1 again. What do you see?*



```
use salesordersexample;

SELECT * FROM Products WHERE ProductNumber=1;
SELECT * FROM Orders WHERE OrderNumber=946;
SELECT * FROM Order_Details WHERE OrderNumber=946;

commit;
```

OrderNumber	ProductNumber	QuotedPrice	QuantityOrdered
946	1	1200.00	2
NULL	NULL	NULL	NULL

Unchanged compared to previous, as commits are visible.

How did the query results differ from the ones in subtask 2.1?

We can now read committed values even if the transaction T1 has not been committed itself.

How can this difference lead to a lost update?

Given two transactions, T1 and T2:

```
X = 100
W1 = +=50
W2 = +=30

T1 ----> R1(X) --------> W1(X) ----------> C1(X)
                       X = 150

T2 -----------> R2(X) --> W2(X) --> C2(X)
                       X = 130
```

T2 is lost as C1(X) commits after C2(X).

Explain the difference in your report and list the necessary SQL statements to produce a lost update at read committed isolation level.

Necessary SQL Statements to create a lost update:

```

-- T1

SELECT QuantityOnHand FROM Products WHERE ProductNumber = 2;

```

```sql

-- T2

SELECT QuantityOnHand FROM Products WHERE ProductNumber = 2;
UPDATE Products SET QuantityOnHand= QuantityOnHand-2 WHERE ProductNumber=1;

COMMIT;
```

```sql

-- T1

UPDATE Products SET QuantityOnHand= QuantityOnHand-5 WHERE ProductNumber=1;

COMMIT;
```

T2 is lost.