Procedure Tests

**GetProductSellPrice**

Data in Database:



Product ID 4, Production\_Cost 10, Margin 3.

Verwachte verkoopprijs zonder enige bonus (karting): 13

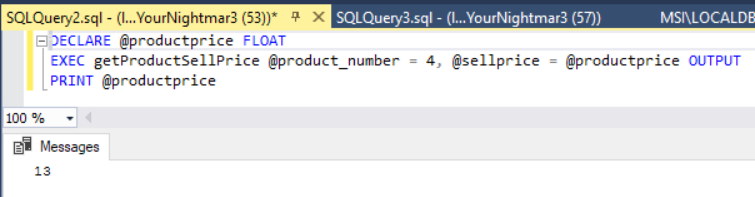
Procedure uitgevoerd zonder korting, maar wel met een historische korting, dus een korting op het product waarvan de datum voorbij is:

Korting:

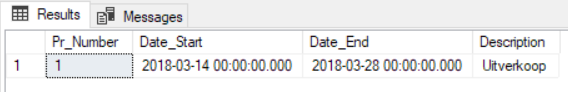


Verwachte uitvoer: 13

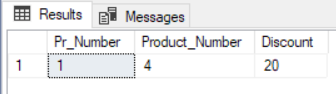
Werkelijke uitvoer:



Dan een test waarbij het einde van de korting in de toekomst ligt, wat dus betekend dat het product op dit moment korting heeft:

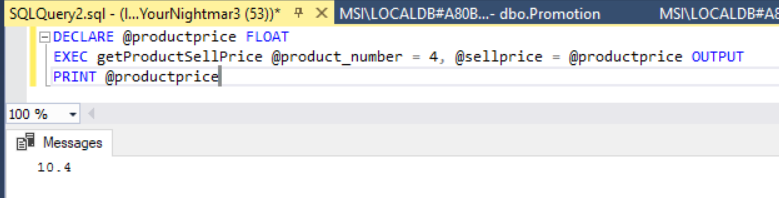


De korting bedraagt 20%:



Verwachte uitvoer: 13\*0.8=10.4

Werkelijke uitvoer:

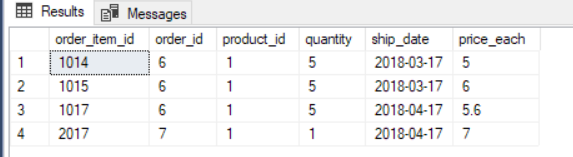


**GetTotalOrderPrice**

Er staat een order in de database:



Bij deze order horen drie items:



De totale prijs van deze order exclusief korting is:

(5\*5)+(5\*6)+(5\*5.6)= 83

De customer had op het moment van het plaatsen van deze order een persoonlijke korting van 5%.

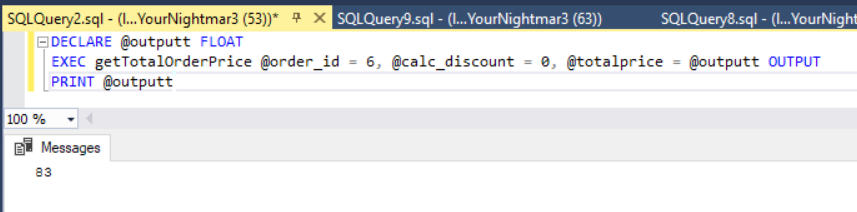
De totale prijs van deze order inclusief korting is:

83\*0.95=78.85

Procedure zonder het berekenen van de korting:

Verwachte uitvoer: 83

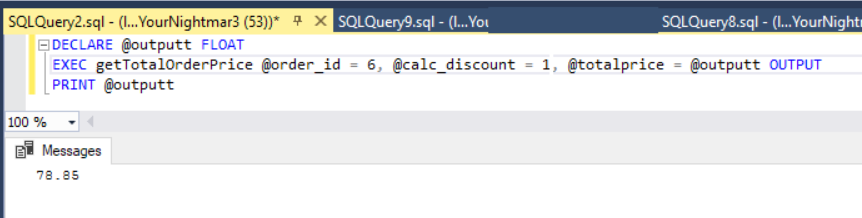
Werkelijke uitvoer:



Procedure met het berekenen van de korting:

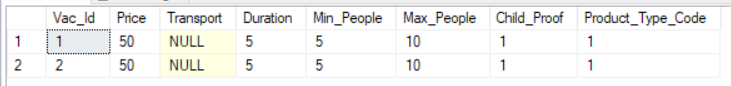
Verwachte uitvoer: 78.85

Werkelijke uitvoer:

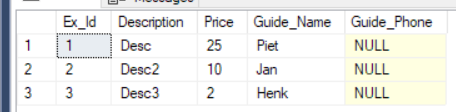


**GetTotalBookingPrice**

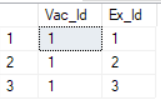
Er zijn twee vakanties in de database:



Ook zijn er drie excursies:

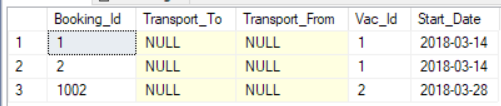


Vakantie ID 1 heeft alle drie de excusies:



Vakantie ID 2 heeft geen excursies.

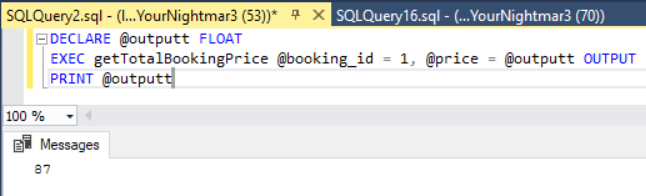
Verder zijn er drie boekingen in de database. Twee boekingen, ID 1 en 2, voor vakantie 1. Een boeking, ID 1002, voor vakantie 2.



De procedure neemt een boeking ID aan en berekent dan de prijs inclusief excursies.

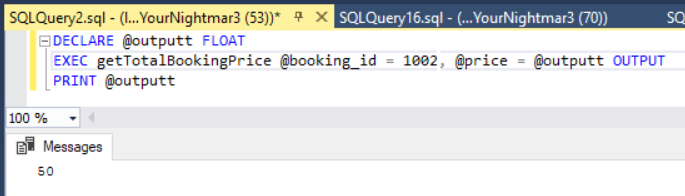
Verwachte uitvoer voor boeking ID 1: 50+25+10+2=87

Werkelijke uitvoer:



Verwachte uitvoer voor boeking ID 1002: 50

Werkelijke uitvoer:



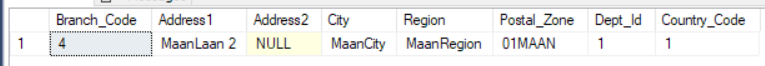
Trigger Tests

**MinimumLoon**

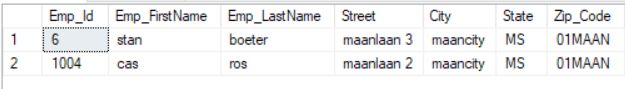
De database heeft een country met een minimumloon van 5.5.



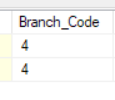
Verder is er een branch in MaanLand:



Er zijn twee employees:

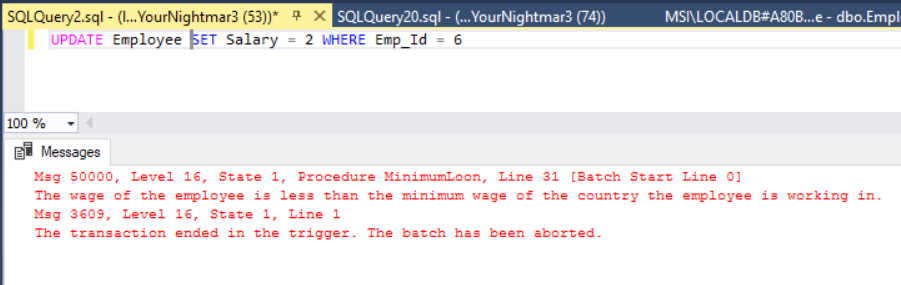


Die allebei in branch code 4 werken, en dus in Country\_Code 1:



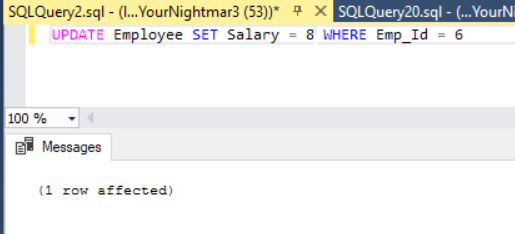
Dit betekent dat het minimumloon van beide employees 5.5 is.

Als we het loon van een employee proberen te veranderen naar iets lager dan 5.5:



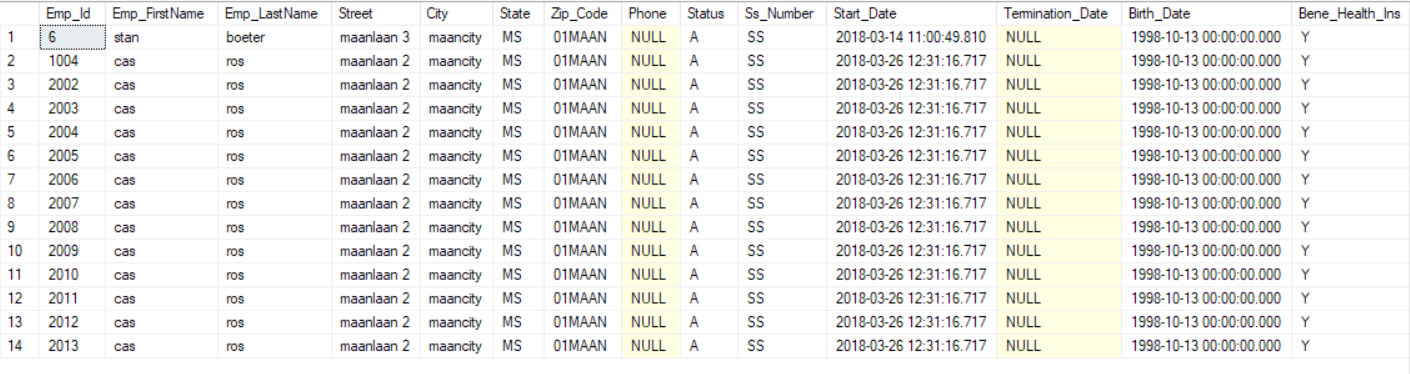
Dan kan dat niet door de trigger.

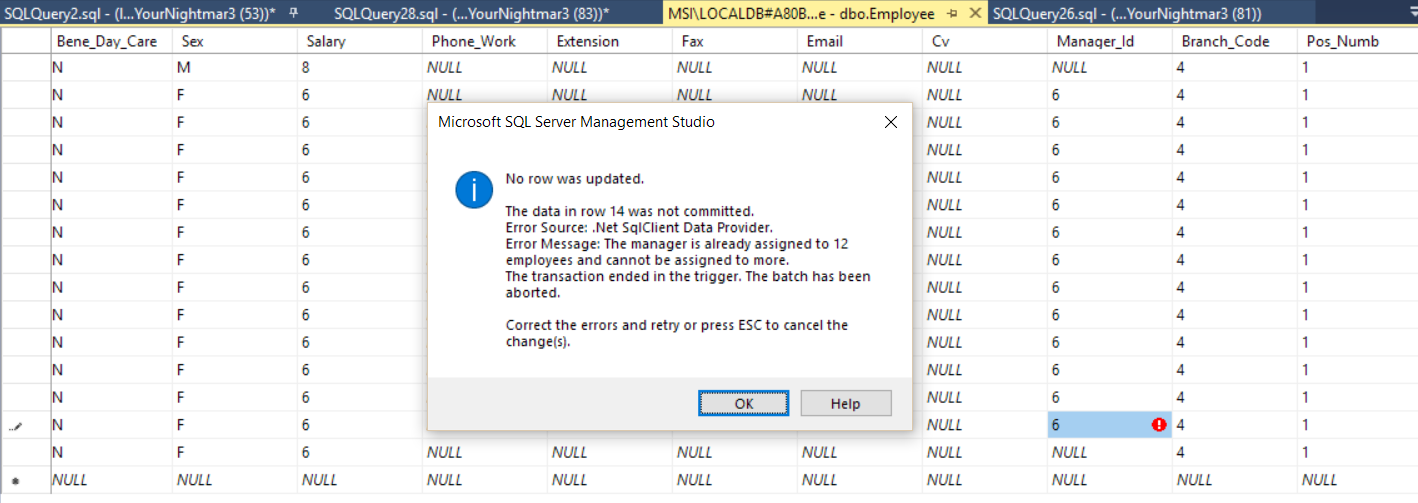
Veranderen naar iets hoger dan het minimumloon kan wel:



**ManagersCheck**

We hebben veertien employees in de database. Hiervan zijn dertien precies het zelfde, maar dat maakt voor de test niets uit. We gaan employee ID 6 als manager gebruiken bij de rest van de employees. Dit zijn er meer dan 12, dus als ik alle employees de manager 6 toewijs, zou de trigger dit proces moeten stoppen.

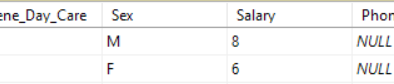


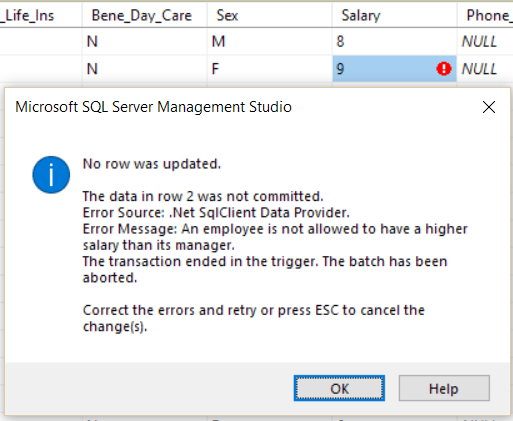


En dit werkt. Op update en insert.

**SalaryCheck**

Er zijn twee employees. De ene employee is de manager van de andere. De manager heeft een salaris van 8. De employee heeft een salaris van 6. Als we het salaris van de employee hoger proberen te maken dan het salaris van zijn manager, zou de trigger dit moeten stoppen.



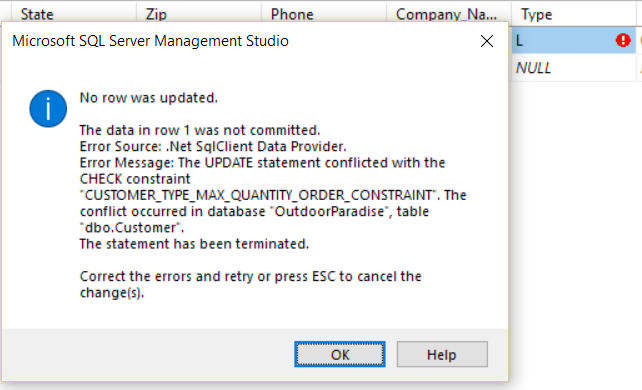


En dit klopt inderdaad. Een werknemer kan nooit een salaris hoger hebben dan zijn manager.

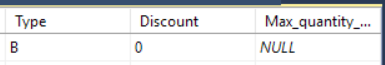
Constraint Tests

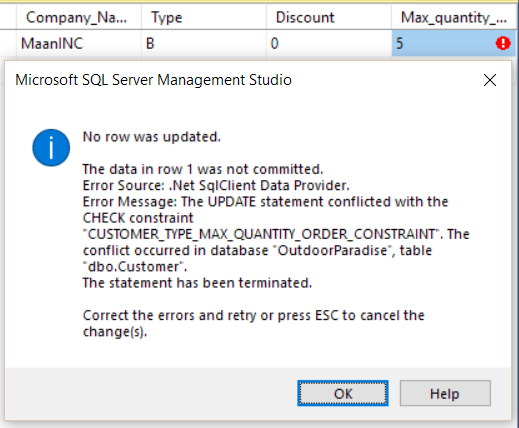
**Constraint CUSTOMER\_TYPE\_MAX\_QUANTITY\_ORDER op de tabel customer.**

Dit is een ingewikkelde constraint, maar is goed te testen. Ten eerste zou dit er voor moeten zorgen dat als we het veld “type” bij een customer veranderen naar iets anders dan een B of een S, dit niet mag.

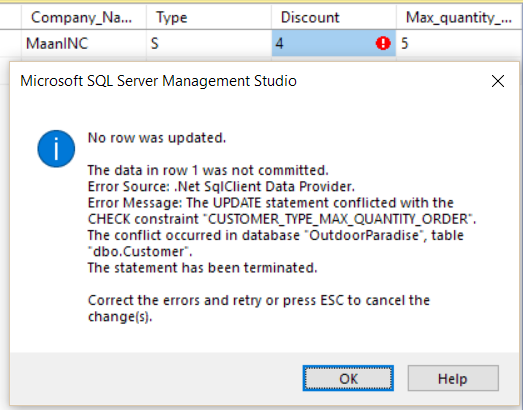


En dat klopt. Verder zou het constraint er voor moeten zorgen dat als type “B” is, max\_quantity\_order NULL moet zijn, en discount hoger of gelijk aan 0 moet zijn.

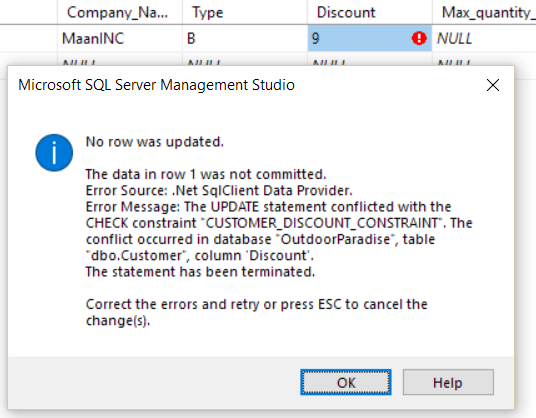




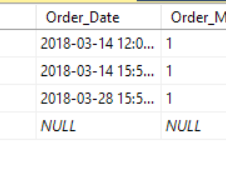
Ook dit klopt volledig. Verder als type “S” is, mag discount alleen 0 zijn, en moet max\_quantity\_order altijd meer dan 0 zijn.



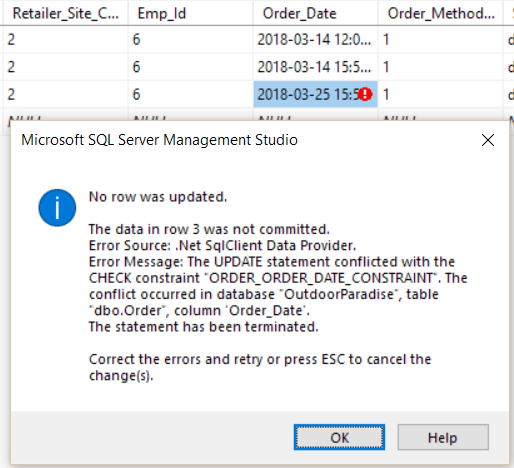
Verder mag Discount sowieso nooit meer dan 8 zijn, omdat dat weer een aparte check is.



**Constraint ORDER\_ORDER\_DATE\_CONSTRAINT op de tabel order**

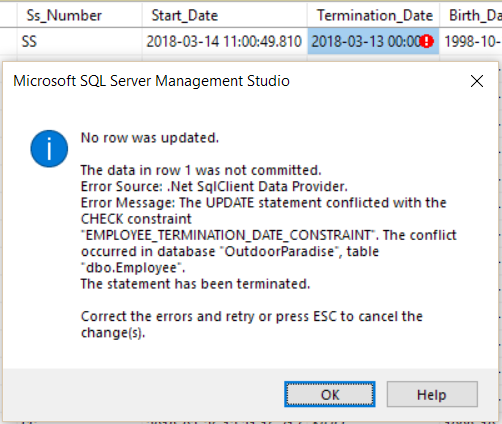


Als we een order proberen aan te maken met een datum in het verleden, dan kan dit niet. De datum op het moment van het schrijven van deze zin is 2018-03-27



**Constraint EMPLOYEE\_TERMINATION\_DATE\_CONSTRAINT op de tabel Employee**

Een employee heeft een start\_date en een termination date. Deze constraint zorgt er voor dat de termination\_date niet voor de start date plaats kan vinden:



Een termination date na de start date mag wel. Ook mag termination date NULL zijn:

