**CREATE** **TABLE** Position **(**

pos\_numb **int** **IDENTITY(**1**,**1**) NOT** **NULL,**

pos\_abbr **Varchar(**20**),**

pos\_title **Varchar(**50**),**

min\_salary **float,**

max\_salary **float,**

training\_id **int,**

**PRIMARY** **KEY** **(**pos\_numb**),**

**FOREIGN** **KEY** **(**training\_id**)** **REFERENCES** Training**(**training\_id**)**

**)**

**CREATE** **TABLE** Target **(**

target\_code **int** **IDENTITY(**1**,**1**) NOT** **NULL,**

target\_description\_en **Varchar(**255**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**target\_code**)**

**)**

**CREATE** **TABLE** Order\_method **(**

order\_method\_code **int** **IDENTITY(**1**,**1**) NOT** **NULL,**

order\_method\_en **Varchar(**255**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**order\_method\_code**)**

**)**

**CREATE** **TABLE** Country **(**

country\_code **int** **IDENTITY(**1**,**1**)** **NOT** **NULL,**

country **Varchar(**50**)** **NOT** **NULL,**

minimum\_wage **Float** **NOT** **NULL** **CHECK([**minimum\_wage**]** **>** 0**)**

**[language]** **Varchar(**50**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**country\_code**)**

**)**

**CREATE** **TABLE** Promotion **(**

pr\_number **Int IDENTITY(**1**,**1**)** **NOT** **NULL,**

date\_start **DateTime,**

date\_end **DateTime,**

**[**description**]** **Varchar(**255**),**

**PRIMARY** **KEY** **(**pr\_number**)**

**)**

**CREATE** **TABLE** Product\_line **(**

prod\_line\_code **int** **IDENTITY(**1**,**1**) NOT** **NULL,**

prod\_line\_en **Varchar(**80**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**prod\_line\_code**)**

**)**

**CREATE** **TABLE** Return\_reason **(**

return\_reason\_code **int** **IDENTITY(**1**,**1**) NOT** **NULL,**

return\_description\_en **Varchar(**50**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**return\_reason\_code**)**

**)**

**CREATE** **TABLE** Retailer\_type **(**

retailer\_type\_code **int** **IDENTITY(**1**,**1**) NOT** **NULL,**

type\_name\_en **Varchar(**255**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**retailer\_type\_code**)**

**)**

**CREATE** **TABLE** Order\_finance **(**

finance\_code **int** **IDENTITY(**1**,**1**)** **NOT** **NULL,**

finance\_type **Varchar(**10**)** **NOT** **NULL** **CHECK([**finance\_type**]** **IN** **(**'expense'**,** 'revenue'**)),**

finance\_description **Varchar(**255**),**

**PRIMARY** **KEY** **(**finance\_code**)**

**)**

**CREATE** **TABLE** Vacation **(**

vac\_id **int** **IDENTITY(**1**,**1**)** **NOT** **NULL,**

transport **Varchar(**255**),**

duration **int** **NOT** **NULL CHECK([**duration**]** **>** 0**),**

price **float** **NOT** **NULL** **CHECK([**price**]** **>** 0**),**

min\_people **int** **NOT** **NULL,**

max\_people **int** **NOT** **NULL,**

child\_proof **bit** **NOT** **NULL,**

**PRIMARY** **KEY** **(**vac\_id**)**

**)**

**CREATE** **TABLE** Bonus **(**

bonus\_date **DATE** **NOT** **NULL,**

bonus\_amo **Float** **NOT** **NULL** **CHECK([**bonus\_amo**]** **>=** 0**),**

emp\_id **int** **NOT** **NULL,**

**CONSTRAINT** **[**PK\_BONUS**]** **PRIMARY** **KEY** CLUSTERED **([**emp\_id**]** **ASC,** **[**bonus\_date**]** **ASC),**

**FOREIGN** **KEY** **(**emp\_id**)** **REFERENCES** Employee**(**emp\_id**)**

**)**

**CREATE** **TABLE** Training **(**

training\_id **int** **IDENTITY(**1**,**1**) NOT** **NULL,**

course **Varchar(**255**),**

pos\_numb **int,**

**PRIMARY** **KEY** **(**training\_id**),**

**FOREIGN** **KEY** **(**pos\_numb**)** **REFERENCES** Training**(**pos\_numb**)**

**)**

**CREATE** **TABLE** Emp\_training**(**

training\_id **int** **NOT** **NULL,**

emp\_id **int** **NOT** **NULL,**

**[date]** **date** **NOT** **NULL**

**CONSTRAINT** **[**EMPLOYEE\_VOLGT**]** **PRIMARY** **KEY** CLUSTERED **([**training\_id**]** **ASC,** **[**emp\_id**]** **ASC,** **[date]** **ASC),**

**FOREIGN** **KEY** **(**training\_id**)** **REFERENCES** Training**(**training\_id**),**

**FOREIGN** **KEY** **(**emp\_id**)** **REFERENCES** Employee**(**emp\_id**)**

**)**

**CREATE** **TABLE** Pref\_appraisal **(**

target\_code **int** **NOT** **NULL,**

**[**description**]** **Varchar(**255**)** **NOT** **NULL,**

manager\_id **int** **NOT** **NULL,**

emp\_id **int** **NOT** **NULL,**

**[timestamp]** **DATETIME** **NOT** **NULL,**

**CONSTRAINT** **[**PK\_PREF\_APPRAISAL**]** **PRIMARY** **KEY** CLUSTERED **([**target\_code**]** **ASC,** **[**manager\_id**]** **ASC,** **[**emp\_id**]** **ASC,** **[Timestamp]** **ASC),**

**FOREIGN** **KEY** **(**emp\_id**)** **REFERENCES** Employee**(**emp\_id**),**

**FOREIGN** **KEY** **(**manager\_id**)** **REFERENCES** Employee**(**emp\_id**)**

**)**

**CREATE** **TABLE** Employee **(**

emp\_id **int** **IDENTITY(**1**,**1**)** **NOT** **NULL,**

emp\_first\_name **Varchar(**20**)** **NOT** **NULL,**

emp\_last\_name **Varchar(**20**)** **NOT** **NULL,**

street **Varchar(**100**),**

city **Varchar(**30**),**

**[state]** **Varchar(**2**),**

zip\_Code **Varchar(**10**),**

phone **Varchar(**20**),**

**[**status**]** **Varchar(**2**),**

ss\_number **Varchar(**10**),**

**[**start\_date**]** **DATETIME** **NOT** **NULL** **DEFAULT** **getdate(),**

termination\_date **DATETIME,**

birth\_date **DATETIME** **NOT** **NULL,**

bene\_health\_ins **Char,**

bene\_life\_ins **Char,**

bene\_day\_care **Char,**

sex **Char** **CHECK(([**sex**]** **IS** **NULL)** **OR** **([**sex**]** **IN** **(**'M'**,** 'F'**))),**

salary **float** **NOT** **NULL** **CHECK([**salary**]** **>** 0**),**

phone\_work **Varchar(**20**),**

extension **Varchar(**6**),**

fax **Varchar(**20**),**

email **Varchar(**50**),**

cv **Varchar(**50**),**

manager\_id **int,**

branch\_code **int,**

pos\_numb **int** **NOT** **NULL,**

**PRIMARY** **KEY** **(**emp\_id**),**

**FOREIGN** **KEY** **(**manager\_id**)** **REFERENCES** Employee**(**emp\_id**),**

**FOREIGN** **KEY** **(**branch\_code**)** **REFERENCES** Branch**(**branch\_code**),**

**FOREIGN** **KEY** **(**pos\_numb**)** **REFERENCES** Position**(**pos\_numb**),**

**CONSTRAINT** **[**EMPLOYEE\_BIRTH\_DATE\_CONSTRAINT**]** **CHECK** **([**Birth\_Date**]<[**Start\_Date**]),**

**CONSTRAINT** **[**EMPLOYEE\_TERMINATION\_DATE\_CONSTRAINT**]** **CHECK** **(([**termination\_date**]** **IS** **NULL)** **OR** **([**Termination\_Date**]** **>** **[**Start\_Date**]))**

**)**

Deze constraints zorgen ervoor dat Termination Date null moet zijn, of na de start date moet plaatsvinden. Ook moet Birth Date voor de Start Date zijn

**CREATE** **TABLE** Department **(**

dept\_id **int** **IDENTITY(**1**,**1**) NOT** **NULL,**

dept\_name **Varchar(**255**)** **NOT** **NULL,**

dept\_head\_id **int** **NOT** **NULL,**

**PRIMARY** **KEY** **(**dept\_id**),**

**FOREIGN** **KEY** **(**dept\_head\_id**)** **REFERENCES** Employee**(**emp\_id**)**

**)**

**CREATE** **TABLE** Branch **(**

branch\_code **int** **IDENTITY(**1**,**1**) NOT** **NULL,**

address1 **Varchar(**50**)** **NOT** **NULL,**

address2 **Varchar(**50**),**

city **Varchar(**40**),**

region **Varchar(**50**),**

postal\_zone **Varchar(**10**),**

dept\_id **int** **NOT** **NULL,**

country\_code **int,**

**PRIMARY** **KEY** **(**branch\_code**),**

**FOREIGN** **KEY** **(**country\_code**)** **REFERENCES** Country**(**country\_code**),**

**FOREIGN** **KEY** **(**dept\_id**)** **REFERENCES** Department**(**dept\_id**)**

**)**

**CREATE** **TABLE** Product\_forecast **(**

product\_number **Int** **NOT** **NULL,**

**[year]** **smallInt** **NOT** **NULL,**

**[month]** **Tinyint** **NOT** **NULL,**

expected\_volume **Int** **CHECK(([**expected\_volume**]** **IS** **NULL)** **OR** **([**expected\_volume**]** **>** 0**)),**

**CONSTRAINT** **[**PK\_PRODUCT\_FORECAST**]** **PRIMARY** **KEY** CLUSTERED **([**product\_number**]** **ASC,** **[Year]** **ASC,** **[Month]** **ASC),**

**FOREIGN** **KEY** **(**product\_number**)** **REFERENCES** Product**(**product\_number**)**

**)**

**CREATE** **TABLE** Storage **(**

storage\_code **Int** **IDENTITY(**1**,**1**) NOT** **NULL,**

city **Varchar(**50**) NOT** **NULL,**

country\_code **int NOT** **NULL,**

**PRIMARY** **KEY** **(**storage\_code**),**

**FOREIGN** **KEY** **(**country\_code**)** **REFERENCES** Country**(**country\_code**)**

**)**

**CREATE** **TABLE** Campaign **(**

pr\_number **Int** **NOT** **NULL,**

product\_number **int** **NOT** **NULL,**

discount **float** **NOT** **NULL** **CHECK** **(([**discount**]** **>=** 0**)** **AND** **([**discount**]** **<=** 100**))**

**CONSTRAINT** **[**PK\_CAMPAIGN**]** **PRIMARY** **KEY** CLUSTERED **([**pr\_number**]** **ASC,** **[**product\_number**]** **ASC),**

**FOREIGN** **KEY** **(**product\_number**)** **REFERENCES** Product**(**product\_number**)**

**)**

**CREATE** **TABLE** Product **(**

product\_number **int** **IDENTITY(**1**,**1**)** **NOT** **NULL,**

introduction\_date **DATE** **NOT** **NULL,**

product\_type\_id **int** **NOT** **NULL,**

production\_cost **Float** **NOT** **NULL** **CHECK** **([**production\_cost**]** **>=** 0**),**

margin **Float** **NOT** **NULL** **CHECK** **([**margin**]** **>=** 0**),**

product\_image **Varchar(**255**)** **NOT** **NULL,**

product\_name **Varchar(**255**),**

product\_description **Varchar(**255**),**

product\_size **Varchar(**20**)** **CHECK([**product\_size**]** **IN** **(**'x-small'**,** 'small'**,** 'medium'**,** 'large'**,** 'x-large'**,** 'xx-large'**,** 'xxx-large'**)),**

product\_color **Varchar(**255**),**

**PRIMARY** **KEY** **(**product\_number**),**

**FOREIGN** **KEY** **(**product\_type\_id**)** **REFERENCES** Product\_type**(**product\_type\_id**)**

**)**

**CREATE** **TABLE** **[Order]** **(**

order\_id **int** **IDENTITY(**1**,**1**)** **NOT** **NULL,**

retailer\_site\_code **int** **NOT** **NULL,**

emp\_id **int** **NOT** **NULL,**

sales\_branch\_code **int** **NOT** **NULL,**

order\_date **DATETIME** **NOT** **NULL** **DEFAULT** **getdate(),**

order\_method\_code **int** **NOT** **NULL,**

finance\_code **int,**

region **Varchar(**50**),**

status **Varchar(**10**)** **NOT** **NULL** **CHECK([**status**]** **=** 'ongoing' **OR** **[**status**]** **=** 'done'**),**

cust\_discount **float** **NOT** **NULL** **DEFAULT** 0 **CHECK(([**cust\_discount**]** **>=** 0**)** **AND** **([**cust\_discount**]** **<=** 8**)),**

**PRIMARY** **KEY** **(**order\_id**),**

**FOREIGN** **KEY** **(**emp\_id**)** **REFERENCES** Employee**(**emp\_id**),**

**FOREIGN** **KEY** **(**order\_method\_code**)** **REFERENCES** Order\_Method**(**order\_method\_code**),**

**FOREIGN** **KEY** **(**retailer\_site\_code**)** **REFERENCES** Retailer\_Site**(**retailer\_site\_code**),**

**FOREIGN** **KEY** **(**finance\_code**)** **REFERENCES** Order\_finance**(**finance\_code**),**

**CONSTRAINT** **[**ORDER\_ORDER\_DATE\_CONSTRAINT**]** **CHECK([**order\_date**]** **>=** **getdate())**

**)**

**CREATE** **TABLE** Customer **(**

id **int** **IDENTITY(**1**,**1**)** **NOT** **NULL,**

fname **Varchar(**255**),**

lname **Varchar(**255**),**

**[**address**]** **Varchar(**255**),**

city **Varchar(**40**),**

**[state]** **Varchar(**255**),**

zip **Varchar(**6**),**

phone **Varchar(**20**),**

company\_name **Varchar(**255**),**

**[type]** **Char** **NOT** **NULL,**

discount **float** **NOT** **NULL** **DEFAULT** 0 **CHECK(([**discount**]** **>=** 0**)** **AND** **([**discount**]** **<=** 8**)),**

max\_quantity\_order **int,**

**PRIMARY** **KEY** **(**id**),**

**CONSTRAINT** **[**CUSTOMER\_TYPE\_MAX\_QUANTITY\_ORDER**]** **CHECK** **((([type]** **=** 'B'**)** **AND** **([**max\_quantity\_order**]** **IS** **NULL)** **AND** **([**discount**]** **>=** 0**))** **OR** **(([type]** **=** 'S'**)** **AND** **([**discount**]** **=** 0**)** **AND** **([**max\_quantity\_order**]** **>** 0**)** **AND** **([**max\_quantity\_order**]** **IS** **NOT** **NULL)))**

**)**

Deze constraint zorgt ervoor dat als het type klant B is, de klant geen max quantity order heeft, en altijd een discount boven 0 heeft. Als het type klant S is, dan is de discount altijd 0, en heeft de klant altijd een max quantity order. Verder moet type altijd B of S zijn en kan het niets anders zijn.

**CREATE** **TABLE** Sales\_target **(**

emp\_id **int** **NOT** **NULL,**

sales\_year **SmallInt** **NOT** **NULL,**

sales\_period **int** **NOT** **NULL,**

retailer\_name **Varchar(**50**)** **NOT** **NULL,**

product\_number **int** **NOT** **NULL,**

sales\_target **Float** **NOT** **NULL** **CHECK([**sales\_target**]** **>** 0**),**

retailer\_code **int** **NOT** **NULL,**

**CONSTRAINT** **[**PK\_SALES\_TARGET**]** **PRIMARY** **KEY** CLUSTERED **([**emp\_id**]** **ASC,** **[**sales\_year**]** **ASC,** **[**sales\_period**]** **ASC),**

**FOREIGN** **KEY** **(**emp\_id**)** **REFERENCES** Employee**(**emp\_id**),**

**FOREIGN** **KEY** **(**product\_number**)** **REFERENCES** Product**(**product\_number**),**

**FOREIGN** **KEY** **(**retailer\_code**)** **REFERENCES** Retailer**(**retailer\_code**)**

**)**

**CREATE** **TABLE** Product\_type **(**

product\_type\_code **int** **IDENTITY(**1**,**1**) NOT** **NULL,**

product\_line\_code **int,**

product\_type\_en **Varchar(**80**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**product\_type\_code**),**

**FOREIGN** **KEY** **(**product\_line\_code**)** **REFERENCES** Product\_Line**(**product\_line\_code**)**

**)**

**CREATE** **TABLE** Order\_item **(**

order\_item\_id **int** **IDENTITY(**1**,**1**)** **NOT** **NULL**

order\_id **int** **NOT** **NULL,**

product\_id **int** **NOT** **NULL,**

quantity **SmallInt** **NOT** **NULL** **DEFAULT** 1 **CHECK([**quantity**]** **>** 0**),**

ship\_date **DATE** **NOT** **NULL,**

price\_each **FLOAT** **NOT** **NULL,**

**PRIMARY** **KEY** **(**order\_item\_id**),**

**FOREIGN** **KEY** **(**order\_id**)** **REFERENCES** **Order(**order\_id**),**

**FOREIGN** **KEY** **(**product\_id**)** **REFERENCES** Product**(**product\_number**)**

**)**

**CREATE** **TABLE** Returned\_item **(**

return\_code **int** **NOT** **NULL,**

return\_date **Datetime** **NOT** **NULL,**

order\_item\_id **int** **NOT** **NULL,**

return\_reason **int** **NOT** **NULL,**

return\_quantity **int** **NOT** **NULL,**

**PRIMARY** **KEY** **(**return\_code**),**

**FOREIGN** **KEY** **(**return\_reason**)** **REFERENCES** Return\_Reason**(**return\_reason\_code**),**

**FOREIGN** **KEY** **(**order\_item\_id**)** **REFERENCES** Order\_item**(**order\_item\_id**)**

**)**

**CREATE** **TABLE** Retailer **(**

retailer\_code **int** **IDENTITY(**1**,**1**)** **NOT** **NULL,**

company\_name **Varchar(**255**)** **NOT** **NULL,**

retailer\_type\_code **int** **NOT** **NULL,**

**PRIMARY** **KEY** **(**retailer\_code**),**

**FOREIGN** **KEY** **(**retailer\_type\_code**)** **REFERENCES** Retailer\_type**(**retailer\_type\_code**)**

**)**

**CREATE** **TABLE** Retailer\_site **(**

retailer\_site\_code **int** **IDENTITY(**1**,**1**)** **NOT** **NULL,**

retailer\_code **int** **NOT** **NULL,**

address1 **Varchar(**50**)** **NOT** **NULL,**

address2 **Varchar(**50**),**

city **Varchar(**40**),**

region **Varchar(**50**),**

postal\_zone **Varchar(**10**),**

country\_code **int NOT NULL,**

active\_indicator **bit NOT NULL,**

**PRIMARY** **KEY** **(**retailer\_site\_code**),**

**FOREIGN** **KEY** **(**country\_code**)** **REFERENCES** Country**(**country\_code**),**

**FOREIGN** **KEY** **(**retailer\_code**)** **REFERENCES** Retailer**(**retailer\_code**)**

**)**

**CREATE** **TABLE** Booking **(**

booking\_id **int** **IDENTITY(**1**,**1**)** **NOT** **NULL,**

transport\_to **Varchar(**255**),**

transport\_from **Varchar(**255**),**

vac\_id **int** **NOT** **NULL,**

**[**start\_date**]** **DATE** **NOT** **NULL,**

**PRIMARY** **KEY** **(**booking\_id**),**

**FOREIGN** **KEY** **(**vac\_id**)** **REFERENCES** Vacation**(**vac\_id**)**

**)**

**CREATE** **TABLE** Traveller **(**

customer\_id **int** **NOT** **NULL,**

booking\_id **int** **NOT** **NULL,**

iBan **Varchar(**255**),**

birth\_date **DATE** **NOT** **NULL,**

sex **Char** **NOT** **NULL,**

**CONSTRAINT** **[**PK\_TRAVELLER**]** **PRIMARY** **KEY** CLUSTERED **([**customer\_id**]** **ASC,** **[**booking\_id**]** **ASC),**

**FOREIGN** **KEY** **(**customer\_id**)** **REFERENCES** Customer**(**customer\_id**),**

**FOREIGN** **KEY** **(**booking\_id**)** **REFERENCES** Booking**(**booking\_id**)**

**)**

**CREATE** **TABLE** Accommodation **(**

acc\_id **int** **IDENTITY(**1**,**1**)** **NOT** **NULL,**

**[**address**]** **Varchar(**255**)** **NOT** **NULL,**

contact **Varchar(**255**),**

phone\_number **Varchar(**20**)** **NOT** **NULL,**

country\_code **int NOT** **NULL,**

**PRIMARY** **KEY** **(**acc\_id**),**

**FOREIGN** **KEY** **(**country\_code**)** **REFERENCES** Country**(**country\_code**)**

**)**

**CREATE** **TABLE** Acc\_vac **(**

acc\_id **int** **NOT** **NULL,**

vac\_id **int** **NOT** **NULL,**

**CONSTRAINT** **[**PK\_ACC\_VAC**]** **PRIMARY** **KEY** CLUSTERED **([**acc\_id**]** **ASC,** **[**vac\_id**]** **ASC),**

**FOREIGN** **KEY** **(**acc\_id**)** **REFERENCES** Accommodation**(**acc\_id**),**

**FOREIGN** **KEY** **(**vac\_id**)** **REFERENCES** Vacation**(**vac\_id**)**

**)**

**CREATE** **TABLE** Excursion **(**

ex\_id **int** **IDENTITY(**1**,**1**)** **NOT** **NULL,**

description **Varchar(**255**),**

price **float** **NOT** **NULL** **CHECK([**price**]** **>** 0**),**

guide\_name **Varchar(**40**)** **NOT** **NULL,**

guide\_phone **Varchar(**40**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**ex\_id**),**

**)**

**CREATE** **TABLE** Vac\_ex **(**

ex\_id **int** **NOT** **NULL,**

vac\_id **int** **NOT** **NULL,**

**CONSTRAINT** **[**PK\_VAC\_EX**]** **PRIMARY** **KEY** CLUSTERED **([**ex\_id**]** **ASC,** **[**vac\_id**]** **ASC),**

**FOREIGN** **KEY** **(**ex\_id**)** **REFERENCES** Excursion**(**ex\_id**),**

**FOREIGN** **KEY** **(**vac\_id**)** **REFERENCES** Vacation**(**vac\_id**)**

**)**

**CREATE** **TABLE** Bo\_ex **(**

ex\_id **int** **NOT** **NULL,**

booking\_id **int** **NOT** **NULL,**

**CONSTRAINT** **[**PK\_BO\_EX**]** **PRIMARY** **KEY** CLUSTERED **([**ex\_id**]** **ASC,** **[**booking\_id**]** **ASC),**

**FOREIGN** **KEY** **(**ex\_id**)** **REFERENCES** Excursion**(**ex\_id**),**

**FOREIGN** **KEY** **(**booking\_id**)** **REFERENCES** Booking**(**booking\_id**)**

**)**

**CREATE** **TABLE** Inventory **(**

inventory\_year **SmallInt** **NOT** **NULL,**

inventory\_month **TinyInt** **NOT** **NULL,**

inventory\_count **int** **NOT** **NULL,**

product\_number **int** **NOT** **NULL,**

**CONSTRAINT** **[**PK\_INVENTORY**]** **PRIMARY** **KEY** CLUSTERED **([**inventory\_year**]** **ASC,** **[**inventory\_month**]** **ASC,** **[**inventory\_count**]** **ASC,** **[**product\_number**]** **ASC),**

**FOREIGN** **KEY** **(**product\_number**)** **REFERENCES** Product**(**product\_number**)**

**)**

Procedures

**GetProductSellPrice**

**CREATE** **PROCEDURE** getProductSellPrice

@product\_number **INT,**

@sellprice **FLOAT** **OUT**

**AS**

**BEGIN**

**SELECT** @discount**=**Discount **FROM** Campaign **JOIN** Promotion **ON** Campaign**.**Pr\_Number **=** Promotion**.**Pr\_Number **WHERE** Promotion**.**Date\_Start **<=** **getdate()** **AND** Promotion**.**Date\_End **>** **getdate()** **AND** Campaign**.**Product\_Number**=**@product\_number

**SELECT** @sellprice**=(**production\_cost **+** margin**)** **FROM** Product **WHERE** product\_number**=**@product\_number

**IF** @discount **IS** **NULL**

**SET** @discount **=** 0

**SET** @sellprice **-=** @sellprice **\*** **(**@discount **/** 100**)**

**END**

Deze procedure berekent de verkoopprijs van een opgegeven product door de productie kosten plus de margin terug te geven. Hier wordt ook rekening gehouden met de korting op een product in de huidige periode.

**CREATE** **PROCEDURE** getTotalOrderPrice**(**

-- Add the parameters for the stored procedure here

@order\_id **INT,**

@calc\_discount **BIT** **=** 0**,**

@totalprice **FLOAT** **OUT**

**)**

**AS**

**DECLARE** @quantity **INT**

**DECLARE** @price\_each **FLOAT**

**SET** @totalprice **=** 0

**DECLARE** the\_cursor **CURSOR** FAST\_FORWARD

**FOR** **SELECT** quantity**,** price\_each **FROM** Order\_item **WHERE** order\_id**=**@order\_id

**OPEN** the\_cursor

**FETCH** **NEXT** **FROM** the\_cursor **INTO** @quantity**,** @price\_each

**WHILE** @@FETCH\_STATUS **=** 0

**BEGIN**

**SET** @totalprice **+=** **(**@price\_each **\*** @quantity**)**

**FETCH** **NEXT** **FROM** the\_cursor **INTO** @quantity**,** @price\_each

**END**

**CLOSE** the\_cursor

**DEALLOCATE** the\_cursor

**IF** **(**@calc\_discount **=** 1**)**

**BEGIN**

**DECLARE** @discount **FLOAT**

**SELECT** @discount**=**cust\_discount **FROM** **[order]** **WHERE** order\_id**=**@order\_id

**SET** @totalprice **-=** @totalprice **\*** **(**@discount**/**100**)**

**END**

Deze procedure berekent de totaalprijs van de opgegeven order. Dit wordt gedaan door van elk item in de order de eerste procedure aan te roepen en de prijs die wordt terug gegeven te vermenigvuldigen met de hoeveelheid die er van het product besteld is. Nadat dit is gedaan wordt de persoonlijke korting van de customer nog toegepast.

**CREATE** **PROCEDURE** getTotalVacationPrice**(**

@vac\_id **INT,**

@price **FLOAT** **OUT**

**)**

**AS**

**BEGIN**

**SELECT** @price**=**price **FROM** vacation **JOIN** Booking **ON** Booking**.**Vac\_Id **=** Vacation**.**Vac\_Id **WHERE** Booking**.**Booking\_Id**=**@booking\_id

**SELECT** @price**+=ISNULL(SUM(**price**),** 0**)** **FROM** Excursion **JOIN** Bo\_ex **AS** Koppel **ON** Koppel**.**ex\_id **=** Excursion**.**Ex\_Id **WHERE** koppel**.**booking\_id**=**@booking\_id

**END**

Deze procedure berekent de prijs van een gegeven boeking. Dit is de prijs van de bijbehorende vakantie plus de prijs van alle excursies die bij de boeking horen. Een vakantie zonder excursies werkt ook.

**CREATE** **PROCEDURE** getTotalSellPriceFromEmployee**(**

@emp\_id **INT,**

@total\_order\_price **FLOAT** **OUT**

**)**

**AS**

**DECLARE** @current\_order\_id **INT**

**SET** @total\_order\_price **=** 0

**DECLARE** the\_cursor **CURSOR** FAST\_FORWARD

**FOR** **SELECT** order\_id **FROM** **[Order]** **WHERE** emp\_id**=**@emp\_id

**OPEN** the\_cursor

**FETCH** **NEXT** **FROM** the\_cursor **INTO** @current\_order\_id

**WHILE** @@FETCH\_STATUS **=** 0

**BEGIN**

**DECLARE** @orderprice **FLOAT**

**EXEC** getTotalOrderPrice @order\_id **=** @current\_order\_id**,** @totalprice **=** @orderprice **OUTPUT**

**SET** @total\_order\_price **+=** @orderprice

**FETCH** **NEXT** **FROM** the\_cursor **INTO** @current\_order\_id

**END**

**CLOSE** the\_cursor

**DEALLOCATE** the\_cursor

Deze procedure berekend de totale prijs van alle orders die bij een employee horen.

Triggers  
**CREATE** **TRIGGER** OutdoorParadise**.**SalaryCheck

**ON** OutdoorParadise**.**Employee

**AFTER** **INSERT,UPDATE**

**AS**

**DECLARE** @emp **FLOAT**

**DECLARE** @man **FLOAT**

**select** @man **=** salary **FROM** inserted**;**

**select** @emp **=** **MAX(**salary**)** **FROM** employee

**WHERE** manager\_id **=** **(select** emp\_id **from** inserted**);**

**IF** @man **<=** @emp

**BEGIN**

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

**SET** NOCOUNT **ON;**

-- Insert statements for trigger here

RAISERROR**(**'The employee is not allowed to be payed more than their manager.'**,** 16**,** 1**);**

**ROLLBACK;**

**RETURN**

**END**

**GO**

Controleert of het salaris van de manager niet lager is dan de ondergeschikten van hem die het meest verdient. Als dit wel zo is verschijnt er een error, en wordt de verandering die voor deze situatie zorgde ongedaan gemaakt.

**CREATE** **TRIGGER** dbo**.**SalaryCheck

**ON** dbo**.**Employee

**AFTER** **INSERT,UPDATE**

**AS**

**DECLARE** @emp **FLOAT**

**DECLARE** @man **FLOAT**

**select** @emp **=** inserted**.**salary **FROM** inserted**;**

**select** @man **=** salary **FROM** employee

**WHERE** emp\_id **=** **(select** manager\_id **from** inserted**);**

**IF** @man **<=** @emp

**BEGIN**

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

**SET** NOCOUNT **ON;**

-- Insert statements for trigger here

RAISERROR**(**'The employee is not allowed to be payed more than their manager.'**,** 16**,** 1**);**

**ROLLBACK;**

**RETURN**

**END**

**GO**

Zorgt ervoor dat het salaris van een werknemer wordt gecontroleerd op het moment dat deze wordt toegevoegd/aangepast. Dit wordt gedaan om te voorkomen dat een werknemer meer kan verdienen dan zijn manager. Als dit wel zo is verschijnt er een error, en wordt de verandering die voor deze situatie zorgde ongedaan gemaakt. Dit was een eis van de opdrachtgever.

**CREATE** **TRIGGER** **[**dbo**].[**Default\_Price\_Each**]**

**ON** **[**dbo**].[**Order\_item**]**

**AFTER** **INSERT**

**AS**

**BEGIN**

**DECLARE** @productnumber **INT**

**SELECT** @productnumber**=**product\_id **FROM** inserted

**DECLARE** @defaultprice **FLOAT**

**EXEC** getProductSellPrice @product\_number **=** @productnumber**,** @sellprice **=** @defaultprice **OUTPUT**

**UPDATE** Order\_item **SET** price\_each**=**@defaultprice **FROM** inserted **WHERE** Order\_Item**.**order\_item\_id **=** inserted**.**order\_item\_id

**END**

Deze trigger zorgt ervoor dat elke keer wanneer er een item aan een order wordt toegevoegd, de huidige prijs van het item inclusief enige bonussen

**CREATE** **TRIGGER** dbo**.**ManagersCheck

**ON** dbo**.**Employee

**AFTER** **INSERT,** **UPDATE**

**AS**

**BEGIN**

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

**SET** NOCOUNT **ON;**

**DECLARE** @manager\_id **INT**

**SELECT** @manager\_id **=** Manager\_Id **FROM** inserted

**IF** @manager\_id **IS** **NOT** **NULL**

**BEGIN**

**DECLARE** @keer **INT**

**SELECT** @keer **=** **COUNT(**emp\_id**)** **FROM** Employee **WHERE** manager\_id **=** @manager\_id**;**

**IF** @keer **>** 12

**BEGIN**

RAISERROR**(**'The manager is already assigned to 12 employees and cannot be assigned to more.'**,** 16**,** 1**);**

**ROLLBACK;**

**RETURN**

**END**

**END**

**END;**

Deze trigger zorgt ervoor dat een manager niet meer dan 12 employees kan hebben. Als dit wel zo is verschijnt er een error, en wordt de verandering die voor deze situatie zorgde ongedaan gemaakt. Dit was een eis van de opdrachtgever.

**CREATE** **TRIGGER** dbo**.**MinimumLoon

**ON** dbo**.**Employee

**AFTER** **INSERT,** **UPDATE**

**AS**

**BEGIN**

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

**SET** NOCOUNT **ON;**

-- Insert statements for trigger here

**DECLARE** @salary **FLOAT**

**DECLARE** @branch\_code **INT**

**SELECT** @salary**=**salary**,** @branch\_code**=**Branch\_Code **FROM** inserted

**IF** @branch\_code **IS** **NOT** **NULL** **AND** @salary **IS** **NOT** **NULL**

**BEGIN**

**DECLARE** @minimum **FLOAT**

**SELECT** @minimum **=** Country**.**Minimum\_Wage

**FROM** Country **JOIN** Branch **ON** Country**.**Country\_code **=** Branch**.**Country\_code

**WHERE** Branch**.**branch\_code **=** @branch\_code

**IF** @salary **<** @minimum

**BEGIN**

RAISERROR**(**'The wage of the employee is less than the minimum wage of the country the employee is working in.'**,** 16**,** 1**);**

**ROLLBACK;**

**RETURN**

**END**

**END**

**END**

Deze trigger zorgt ervoor dat het salaris van een employee niet minder kan zijn dan het minimum salaris van het land van de branch waar de employee in werkt. Als dit wel zo is verschijnt er een error, en wordt de verandering die voor deze situatie zorgde ongedaan gemaakt.