



Wrocław University
of Science and Technology

Data Warehouses

Laboratory 4

AleksandraTrojok 245670

Waheeda Kadar Rashiya 245807

Task 1

Analyze sales data based on the color of product and class. This query gives a visualization of how the product color and class affect different sales data shown below. Left join would give null values of some sales data or example for product colored grey and class not assigned and we want only the products, which were sold.

```
SELECT COALESCE(pr.Color, 'No color') AS Color, COALESCE(pr.Class, 'Not assigned') AS  
Class, sum(sd.OrderQty) as Quantity, sum (sd.LineTotal) as SalesValue, sum  
(sd.LineTotal-sd.OrderQty*StandardCost) AS Profit, sum(sh.Freight) as  
shipmentCosts, sum(sd.UnitPriceDiscount ) as Discounts
```

```
FROM Production.Product pr
```

```
join Sales.SalesOrderHeader sh
```

```
join sales.SalesOrderDetail sd
```

```
ON sd.SalesOrderID=sh.SalesOrderID
```

```
on pr.ProductID=sd.ProductID
```

```
group by Color, Class
```

Color	Class	Quantity	SalesValue	Profit	shipmentCosts	Discounts
Multi	Not assigned	25073	649849.160346	-48604.382554	5709781.4066	8.62
Silver/Black	Not assigned	147	7143.318000	1857.256800	107338.3538	0.00
Blue	M	2690	2133111.866340	101756.214340	1034954.0467	0.34
Red	Not assigned	6266	157772.394392	75773.638592	1157186.4307	8.57
Blue	L	3153	1232481.531390	-73084.377110	1544387.0025	34.02
Red	H	7024	14243669.146325	2513218.384125	2921232.2293	0.11
Yellow	M	15184	11587643.955804	273632.200804	5898574.8697	1.68
No color	L	4151	111328.211316	44014.296916	1299752.6025	0.11
Silver/Black	M	1006	37477.524000	9744.116000	484537.5646	0.00
No color	M	4829	164091.767497	61469.058097	1023133.1478	0.38
Silver/Black	L	1317	31995.198000	8318.698800	540859.509	0.00
Black	M	6888	2612496.496430	259430.576730	2914194.1388	0.33
Silver	H	11560	17123491.101956	2217624.650156	4303103.6539	24.35
Yellow	H	4265	5250897.304655	-103994.801445	1728857.5309	46.70
Black	Not assigned	24100	1021121.005500	272274.190400	6022706.1641	29.61

Query executed successfully. | DESKTOP-GHPLDUQ (14.0 RTM) | DESKTOP-GHPLDUQ\Ola (52) | AdventureWorks2017 | 00:00:00 | 30 rows

analyze sales for subcategory and difference in orderdate nad shipdate

```

SELECT ps.Name as Subcategory ,avg(DATEDIFF(DAY,OrderDate,ShipDate) )as AvgDifference,
sum(ss.OrderQty) as Quantity,sum (ss.LineTotal) as SalesValue,sum (ss.LineTotal-
ss.OrderQty*StandardCost) AS Profit,sum(sh.Freight) as
shipmentCosts,sum(ss.UnitPriceDiscount ) as Discounts

FROM Production.ProductSubcategory ps

JOIN Production.Product pp

on ps.ProductSubcategoryID=pp.ProductSubcategoryID

JOIN Sales.SalesOrderDetail ss

on ss.ProductID=pp.ProductID

INNER JOIN Sales.SalesOrderHeader sh

on ss.SalesOrderID=sh.SalesOrderID

group by ps.Name

```

Subcategory	AvgDifference	Quantity	SalesValue	Profit	shipmentCosts	Discounts
Socks	7	5217	29745.127584	12163.310484	817419.4633	3.16
Cleaners	7	3319	18406.972080	8538.589380	493943.0787	1.63
Saddles	7	2145	55829.388248	14476.554448	1177994.703	0.08
Deraillleurs	7	1166	70209.495800	18147.480400	494781.6062	0.08
Touring Bikes	7	14751	14296291.259139	217277.703139	5721917.7891	123.39
Hydration Packs	7	2761	105826.418334	49042.864034	451406.5696	1.15
Bike Racks	7	3166	237096.156000	95006.076000	488727.9396	2.09
Mountain Frames	7	11621	4713930.229240	261604.802740	6015192.3691	0.30
Brakes	7	1035	66018.711000	17077.701000	443343.4673	0.06
Jerseys	7	22711	752259.388034	-149887.157266	4725530.3534	9.92
Bike Stands	7	249	39591.000000	24783.966000	3751.2329	0.00
Chains	7	774	9377.710144	2422.081744	277966.5633	0.06
Mountain Bikes	7	28321	36445443.937380	4908041.892480	9581534.7709	117.06
Tights	7	4589	203149.079844	61195.707244	1220539.9008	1.94
Bike Shorts	7	3126	167659.617207	51555.904807	1002191.200	0.60

Query executed successfully. | DESKTOP-GHPLDUQ (14.0 RTM) | DESKTOP-GHPLDUQ\Ola (52) | AdventureWorks2017 | 00:00:00 | 35 rows

Analyze sales for colors

```

SELECT COALESCE(pr.Color, 'No color') AS Color ,sum(sd.OrderQty) as Quantity,sum
(sd.LineTotal) as SalesValue,sum (sd.LineTotal-sd.OrderQty*StandardCost) AS
Profit,sum(sh.Freight) as shipmentCosts,sum(sd.UnitPriceDiscount ) as Discounts

FROM Production.Product pr

```

```

join Sales.SalesOrderHeader sh

join sales.SalesOrderDetail sd

ON sd.SalesOrderID=sh.SalesOrderID

on pr.ProductID=sd.ProductID


group by Color

```

Color	Quantity	SalesValue	Profit	shipmentCosts	Discounts
Silver	25023	19777339.945262	2622674.676262	9456472.7396	93,56
No color	48289	1099318.907552	461119.427152	7301955,618	6,27
Red	29229	21623269.540854	2280584.251154	10064674,6929	28,89
Multi	25073	649849.160346	-48604.382554	5709781,4066	8,62
Black	81937	38247018.631195	3399774.338795	26027435,8234	55,78
Blue	23659	9602850.958400	606570.442700	6596596,4341	50,72
Yellow	32556	18669505.218895	-629.403405	11026174,8201	95,77
White	5217	29745.127584	12163.310484	817419,4633	3,16
Silver/Black	3931	147483.909800	38250.965800	1689887.2051	0.08

Analyze sales data based on the color of product and class. We have used join to eliminate null values from data sales. Left join, because not all categories might have subcategories in this case they do. This query give us all combination for categories and their matching subcategories with the information on sales.

```

SELECT c.Name, sc.Name, sum(sd.OrderQty) as Quantity, sum (sd.LineTotal) as
SalesValue, sum (sd.LineTotal-sd.OrderQty*StandardCost) AS Profit, sum(sh.Freight) as
shipmentCosts, sum(sd.UnitPriceDiscount ) as Discounts

FROM Production.Product pr

join Sales.SalesOrderHeader sh

join sales.SalesOrderDetail sd

ON sd.SalesOrderID=sh.SalesOrderID

```

```

on pr.ProductID=sd.ProductID

left join Production.ProductSubcategory sc

on sc.ProductSubcategoryID=pr.ProductSubcategoryID

join Production.ProductCategory c

ON sc.ProductCategoryID=c.ProductCategoryID

group by c.Name, sc.Name

```

Name	Name	Quantity	SalesValue	Profit	shipmentCosts	Discounts
Clothing	Vests	6738	259488.368500	99467.606500	963563.9336	6.53
Components	Brakes	1035	66018.711000	17077.701000	443343.4673	0.06
Components	Wheels	5273	680831.354061	176044.865961	2526215.0972	0.30
Components	Headsets	1009	60942.198385	15448.045685	358207.6156	0.24
Components	Road Frames	11753	3851350.600747	-172617.287753	6467854.5793	0.21
Accessories	Fenders	2121	46619.580000	29183.899500	48103.2053	0.00
Clothing	Bib-Shorts	3125	167558.617307	51555.804807	1092191.308	0.60
Accessories	Tires and Tubes	18006	246454.527632	154048.677832	421342.246	0.02
Clothing	Caps	8311	51229.445623	-6301.789677	1332498.7323	3.60
Bikes	Mountain Bikes	28321	36445443.937380	4908041.892480	9581534.7709	117.06
Components	Saddles	2145	55829.388248	14476.554448	1177994.703	0.08
Clothing	Shorts	9967	413600.513342	155974.086442	1801587.7426	8.79
Components	Derailleurs	1166	70209.495800	18147.480400	494781.6062	0.08
Components	Chains	774	93777.710144	2422.081744	277966.5633	0.06
Components	Bottom Brackets	921	51826.274000	12474.820400	482860.2786	0.00

ery executed successfully. | DESKTOP-GHPLDUQ (14.0 RTM) | DESKTOP-GHPLDUQ\Ola (52) | AdventureWorks2017 | 00:00:00 | 35 rows

Analyze sales data based on different branches

Sales data for a department

```

SELECT dp.Name ,sum(sd.OrderQty) as Quantity,sum (sd.LineTotal) as SalesValue,
(sum(sd.LineTotal)-sum(sd.OrderQty*StandardCost)) AS Profit,sum(sh.Freight) as
shipmentCosts,sum(sd.UnitPriceDiscount ) as Discounts

FROM Sales.SalesOrderHeader sh

join sales.SalesOrderDetail sd

ON sd.SalesOrderID=sh.SalesOrderID

```

```

join Production.Product pr

on pr.ProductID=sd.ProductID

join Sales.SalesPerson sp

on sp.BusinessEntityID=sh.SalesPersonID

join HumanResources.EmployeeDepartmentHistory dh

on sp.BusinessEntityID=dh.BusinessEntityID

join HumanResources.Department dp

on dh.DepartmentID=dp.DepartmentID

group by dp.Name

```

Name	Quantity	SalesValue	Profit	shipmentCosts	Discounts
Sales	214516	80487704.179188	-2316039.253112	77209717,2137	342,85

Analyze data for different years of orders

```

SELECT Year(OrderDate) as OrderYears ,sum(sd.OrderQty) as Quantity,sum (sd.LineTotal)
as SalesValue,sum (sd.LineTotal-sd.OrderQty*StandardCost) AS Profit,sum(sh.Freight) as
shipmentCosts,sum(sd.UnitPriceDiscount ) as Discounts

FROM Production.Product pr

join Sales.SalesOrderHeader sh

join sales.SalesOrderDetail sd

ON sd.SalesOrderID=sh.SalesOrderID

on pr.ProductID=sd.ProductID

group by Year(OrderDate)

order by Year(OrderDate)Desc

```

OrderYears	Quantity	SalesValue	Profit	shipmentCosts	Discounts
2014	61659	20057928.810865	3442493.663265	11710848,7257	78,64
2013	131788	43622479.051635	3354362.929135	36248928,0339	172,33
2012	68579	33524301.324434	948201.191134	26188334,8687	90,91
2011	12888	12641672.212954	1626845.842854	4542286,5748	0,97

Analyze data for different months and year of orders

```

SELECT Year(OrderDate) as OrderYears ,Month(OrderDate) as
OrderMonths,sum(sd.OrderQty) as Quantity,sum (sd.LineTotal) as SalesValue,sum
(sd.LineTotal-sd.OrderQty*StandardCost) AS Profit,sum(sh.Freight) as
shipmentCosts,sum(sd.UnitPriceDiscount ) as Discounts

FROM Production.Product pr

join Sales.SalesOrderHeader sh

```

```

join sales.SalesOrderDetail sd

ON sd.SalesOrderID=sh.SalesOrderID

on pr.ProductID=sd.ProductID

group by Year(OrderDate), Month(OrderDate)

order by Year(OrderDate)Desc, Month(OrderDate)

```

OrderYears	OrderMonths	Quantity	SalesValue	Profit	shipmentCosts	Discounts
2014	1	11463	4289817.950953	647468.820553	2119240.1659	1.05
2014	2	4287	1337725.035600	553029.957400	92653.4597	0.00
2014	3	22582	7217531.091974	675665.151674	5752956.8759	42.81
2014	4	5313	1797173.923000	746490.173900	125979.19	0.00
2014	5	15884	5366674.969338	792473.558238	3616712.4228	34.78
2014	6	2130	49005.840000	27366.001500	3306.6114	0.00
2013	1	4154	2087872.462504	131668.842404	1216264.5044	0.33
2013	2	5651	2316922.151480	99385.045080	2173610.6449	1.49
2013	3	8250	3412068.967535	65022.944035	3154447.8585	1.52
2013	4	6449	2532265.912399	133866.489799	2186007.0687	1.49
2013	5	10260	3245623.754479	29385.881679	3543718.7239	43.16
2013	6	16611	5081069.131596	102147.019596	5391230.7661	46.40
2013	7	18589	4896353.737794	75823.258594	5339338.2316	65.69
2013	8	11548	3333964.067555	434840.100655	2500023.2854	4.02
2012	9	14576	4522909.705202	456772.269002	3569756.6928	2.42

Query executed successfully. | DESKTOP-GHPLDUQ (14.0 RTM) | DESKTOP-GHPLDUQ\Ola (52) | AdventureWorks2017 | 00:00:00 | 38 rows

Analyze Different individual customers (names, addresses)

```

SELECT sc.CustomerID,pp.FirstName as FirstName, pa.AddressLine1 as AddressR,

sum(sd.OrderQty) as Quantity,sum (sd.LineTotal) as SalesValue,sum (sd.LineTotal-
sd.OrderQty*StandardCost) AS Profit,sum(sh.Freight) as
shipmentCosts,sum(sd.UnitPriceDiscount ) as Discounts

```

```

FROM Sales.SalesOrderHeader sh

join sales.SalesOrderDetail sd

ON sd.SalesOrderID=sh.SalesOrderID

join Production.Product pr

on pr.ProductID=sd.ProductID

join Sales.Customer sc

```



```

on sh.CustomerID=sc.CustomerID

join Person.Person pp on sc.PersonID=pp.BusinessEntityID

INNER JOIN Person.Address pa

ON pa.AddressID=sh.BillToAddressID

where PersonType = 'IN'

group by sc.CustomerID,pa.AddressLine1, pp.FirstName

```

CustomerID	FirstName	AddressR	Quantity	SalesValue	Profit	shipmentCosts	Discounts
29483	Jésus	244, rue de la Centenaire	1	2049.098200	797.116900	51.2275	0.00
29482	Clayton	1080, quai de Grenelle	1	2049.098200	797.116900	51.2275	0.00
29481	Ivan	Knaackstr 4	1	3374.990000	1476.895600	84.3748	0.00
29480	Nina	9 Katherine Drive	5	2442.030000	934.854900	305.254	0.00
29479	Tommy	111, rue Maillard	1	2049.098200	797.116900	51.2275	0.00
29478	Darren	5240 Premier Pl.	3	2398.050000	910.883500	179.8539	0.00
29477	Neil	P.O. Box 9178	3	2428.050000	926.103500	182.1039	0.00
29476	Elizabeth	Nonnendamm 2	1	3399.990000	1487.835600	84.9998	0.00
29475	Jared	Ertplatz 876	1	3399.990000	1487.835600	84.9998	0.00
29474	Jaime	Potsdamer Straße 646	1	3374.990000	1476.895600	84.3748	0.00
29473	Camren	6467 Buena Vista	2	2419.060000	924.035800	120.953	0.00
29472	Lacey	21, avenue Reille	3	88.980000	55.701400	6.6735	0.00
29471	Dana	80, rue de Fontfroide	2	23.780000	14.886200	1.189	0.00
29470	Nathan	29, boulevard Beau Marchais	3	60.470000	37.854100	4.5354	0.00
29469	Dominique	2782 Westwood Dr	2	2282.060000	987.768800	119.652	0.00

Query executed successfully. | DESKTOP-GHPLDUQ (14.0 RTM) | DESKTOP-GHPLDUQ\Ola (56) | AdventureWorks2017 | 00:00:01 | 18484 rows

Task 2.1

```
DECLARE @StartDate DATE = '20100101', @NumberOfYears INT = 11;
```

```
SET DATEFIRST 7;
```

```
SET DATEFORMAT mdy;
```

```
SET LANGUAGE US_ENGLISH;
```

```
DECLARE @CutoffDate DATE = DATEADD(YEAR, @NumberOfYears, @StartDate);
```

```
CREATE TABLE dimDate
```

```
(  
    [date]          DATE PRIMARY KEY,  
    [day]           AS DATEPART(DAY, [date]),  
    [month]         AS DATEPART(MONTH, [date]),  
    FirstOfMonth AS CONVERT(DATE, DATEADD(MONTH, DATEDIFF(MONTH, 0, [date]), 0)),  
    [MonthName]     AS DATENAME(MONTH, [date]),  
    [week]          AS DATEPART(WEEK, [date]),  
    [ISOweek]       AS DATEPART(ISO_WEEK, [date]),  
    [DayOfWeek]     AS DATEPART(WEEKDAY, [date]),  
    [DayName]       AS DATENAME(WEEKDAY, [date]),  
    [quarter]       AS DATEPART(QUARTER, [date]),  
    [year]          AS DATEPART(YEAR, [date]),  
    EUStyle         AS CONVERT(varchar, [date], 105),  
    IsWeekend       AS CONVERT(BIT, CASE WHEN DATEPART(WEEKDAY, [date]) IN (1,7) THEN 1  
ELSE 0 END),  
    FirstOfYear     AS CONVERT(DATE, DATEADD(YEAR, DATEDIFF(YEAR, 0, [date]), 0)),  
    DateID          AS CONVERT(CHAR(8), [date], 112),  
    --PRIMARY KEY(DateID, Date)  
    Style101        AS CONVERT(CHAR(10), [date], 101)  
);
```

```
INSERT dimDate([date])
```

```
SELECT d
```

```
FROM
```

```
(
SELECT d = DATEADD(DAY, rn - 1, @StartDate)

FROM

(
SELECT TOP (DATEDIFF(DAY, @StartDate, @CutoffDate))

rn = ROW_NUMBER() OVER (ORDER BY s1.[object_id])

FROM sys.all_objects AS s1

CROSS JOIN sys.all_objects AS s2

ORDER BY s1.[object_id]

) AS x
```

dim.date Result:

date	day	month	FirstOfMonth	MonthName	week	ISOweek	DayOfWeek	DayName	quarter	year	EUStyle	IsWeekend	FirstOfYear	DateID	Style101
2010-01-01	1	1	2010-01-01	January	1	53	6	Friday	1	2010	01-01-2010	0	2010-01-01	20100101	01/01/2010
2010-01-02	2	1	2010-01-01	January	1	53	7	Saturday	1	2010	02-01-2010	1	2010-01-01	20100102	01/02/2010
2010-01-03	3	1	2010-01-01	January	2	53	1	Sunday	1	2010	03-01-2010	1	2010-01-01	20100103	01/03/2010
2010-01-04	4	1	2010-01-01	January	2	1	2	Monday	1	2010	04-01-2010	0	2010-01-01	20100104	01/04/2010
2010-01-05	5	1	2010-01-01	January	2	1	3	Tuesday	1	2010	05-01-2010	0	2010-01-01	20100105	01/05/2010
2010-01-06	6	1	2010-01-01	January	2	1	4	Wednesday	1	2010	06-01-2010	0	2010-01-01	20100106	01/06/2010
2010-01-07	7	1	2010-01-01	January	2	1	5	Thursday	1	2010	07-01-2010	0	2010-01-01	20100107	01/07/2010
2010-01-08	8	1	2010-01-01	January	2	1	6	Friday	1	2010	08-01-2010	0	2010-01-01	20100108	01/08/2010
2010-01-09	9	1	2010-01-01	January	2	1	7	Saturday	1	2010	09-01-2010	1	2010-01-01	20100109	01/09/2010
2010-01-10	10	1	2010-01-01	January	3	1	1	Sunday	1	2010	10-01-2010	1	2010-01-01	20100110	01/10/2010
2010-01-11	11	1	2010-01-01	January	3	2	2	Monday	1	2010	11-01-2010	0	2010-01-01	20100111	01/11/2010
2010-01-12	12	1	2010-01-01	January	3	2	3	Tuesday	1	2010	12-01-2010	0	2010-01-01	20100112	01/12/2010
2010-01-13	13	1	2010-01-01	January	3	2	4	Wednesday	1	2010	13-01-2010	0	2010-01-01	20100113	01/13/2010
2010-01-14	14	1	2010-01-01	January	3	2	5	Thursday	1	2010	14-01-2010	0	2010-01-01	20100114	01/14/2010
2010-01-15	15	1	2010-01-01	January	3	2	6	Friday	1	2010	15-01-2010	0	2010-01-01	20100115	01/15/2010
2010-01-16	16	1	2010-01-01	January	3	2	7	Saturday	1	2010	16-01-2010	1	2010-01-01	20100116	01/16/2010
2010-01-17	17	1	2010-01-01	January	4	2	1	Sunday	1	2010	17-01-2010	1	2010-01-01	20100117	01/17/2010
2010-01-18	18	1	2010-01-01	January	4	3	2	Monday	1	2010	18-01-2010	0	2010-01-01	20100118	01/18/2010
2010-01-19	19	1	2010-01-01	January	4	3	3	Tuesday	1	2010	19-01-2010	0	2010-01-01	20100119	01/19/2010
2010-01-20	20	1	2010-01-01	January	4	3	4	Wednesday	1	2010	20-01-2010	0	2010-01-01	20100120	01/20/2010
2010-01-21	21	1	2010-01-01	January	4	3	5	Thursday	1	2010	21-01-2010	0	2010-01-01	20100121	01/21/2010
2010-01-22	22	1	2010-01-01	January	4	3	6	Friday	1	2010	22-01-2010	0	2010-01-01	20100122	01/22/2010
2010-01-23	23	1	2010-01-01	January	4	3	7	Saturday	1	2010	23-01-2010	1	2010-01-01	20100123	01/23/2010

Task 2.2

1. Checking the total number of rows:

```
SELECT COUNT(date) AS NoOfRows FROM dimDate
```

Result:

```
NoOfRows
```

```
-----
```

```
4018
```

```
(1 row affected)
```

2. Number of unique years:

```
SELECT COUNT(DISTINCT year) As NoOfUniqueYears FROM dimDate;
```

Result:

```
NoOfUniqueYears
-----
11

(1 row affected)|
```

3. Unique year-month combinations:

```
SELECT COUNT(*) FROM (SELECT DISTINCT year, month FROM dimDate) AS diff;
```

Result:

```
-----
132

(1 row affected)
```

4. 5 random rows for logical business correctness:

```
select top 5* from dimDate order by date
```

Result:

date	day	month	FirstOfMonth	MonthName	week	ISOweek	DayOfWeek	DayName	quarter	year	EUSStyle	IsWeekend	FirstOfYear	DateID	Style101
2010-01-01	1	1	2010-01-01	January	1	53	6	Friday	1	2010	01-01-2010	0	2010-01-01	20100101	01/01/2010
2010-01-02	2	1	2010-01-01	January	1	53	7	Saturday	1	2010	02-01-2010	1	2010-01-01	20100102	01/02/2010
2010-01-03	3	1	2010-01-01	January	2	53	1	Sunday	1	2010	03-01-2010	1	2010-01-01	20100103	01/03/2010
2010-01-04	4	1	2010-01-01	January	2	1	2	Monday	1	2010	04-01-2010	0	2010-01-01	20100104	01/04/2010
2010-01-05	5	1	2010-01-01	January	2	1	3	Tuesday	1	2010	05-01-2010	0	2010-01-01	20100105	01/05/2010

(5 rows affected)

Task 2.3

a) `SELECT` pp.ProductID, pp.Name, pp.ProductNumber, `COALESCE`(pps.Name, 'no subcat') `AS` Subcategory, `COALESCE`(ppc.Name, 'no category') `AS` Category, pp.ListPrice, `COALESCE`(pp.Color, 'No color') `AS` Color, pp.ProductLine, pp.Class, pp.Weight, pp.Size,

`CASE`

`WHEN`

`(`

`SELECT DISTINCT` ProductID

`FROM`

AdventureWorks2017.Production.TransactionHistory pth

`WHERE` pp.ProductID = pth.ProductID

`) IS NULL THEN 'Not purchased'`

`ELSE 'Purchased'`

```

        END

        AS Purchased,

CASE pp.DaysToManufacture

        WHEN(

                0

        ) THEN 'Manufactured'

        ELSE 'Not Manufactured'

        END

        AS Manufactured

    INTO dwaw.DIMProduct

FROM AdventureWorks2017.Production.Product pp

LEFT JOIN AdventureWorks2017.Production.ProductSubcategory pps

        ON pps.ProductSubcategoryID = pp.ProductSubcategoryID

LEFT JOIN AdventureWorks2017.Production.ProductCategory ppc

        ON ppc.ProductCategoryID = pps.ProductCategoryID

```

For this task we have selected all the given in description columns, baring in mind that we need to check whether a product has been purchased at least once and whether it has been manufactured.-Number of days required to manufacture the product is 0 if it has not yet been manufactured. In Production.TransactionHistory ProductID is a foreign key to Product.ProductID., if it is null it means product has not been purchased. In our understanding , the purpose of this table dimProduct was to show details of all productIDs which exist in Production.Product, for important information like color , category or subcategory we have put NULL values to as names like ‘No

color', 'no subcat', 'no category', for the less important we have left the NULL values. Usage of left join is because not all products have categories and not all categories have subcategories, but we still want other valuable information about existing products.

ProductID	Name	ProductNumber	Subcategory	Category	ListPrice	Color	ProductLine	Class	Weight	Size	Purchased	Manufactured
1	Adjustable Race	AR-5381	no subcat	no category	0.00	No color	NULL	NULL	NULL	NULL	Purchased	Manufactured
2	Bearing Ball	BA-8327	no subcat	no category	0.00	No color	NULL	NULL	NULL	NULL	Purchased	Manufactured
3	BB Ball Bearing	BE-2349	no subcat	no category	0.00	No color	NULL	NULL	NULL	NULL	Purchased	Not Manufactured
4	Headset Ball Bearings	BE-2908	no subcat	no category	0.00	No color	NULL	NULL	NULL	NULL	Purchased	Manufactured
316	Blade	BL-2036	no subcat	no category	0.00	No color	NULL	NULL	NULL	NULL	Purchased	Not Manufactured
317	LL Crankarm	CA-5965	no subcat	no category	0.00	Black	NULL	L	NULL	NULL	Purchased	Manufactured
318	ML Crankarm	CA-6738	no subcat	no category	0.00	Black	NULL	M	NULL	NULL	Purchased	Manufactured
319	HL Crankarm	CA-7457	no subcat	no category	0.00	Black	NULL	NULL	NULL	NULL	Purchased	Manufactured
320	Chaining Bolts	CB-2903	no subcat	no category	0.00	Silver	NULL	NULL	NULL	NULL	Purchased	Manufactured
321	Chaining Nut	CN-6137	no subcat	no category	0.00	Silver	NULL	NULL	NULL	NULL	Purchased	Manufactured
322	Chaining	CR-7833	no subcat	no category	0.00	Black	NULL	NULL	NULL	NULL	Purchased	Manufactured
323	Crown Race	CR-9981	no subcat	no category	0.00	No color	NULL	NULL	NULL	NULL	Purchased	Manufactured
324	Chain Stays	CS-2812	no subcat	no category	0.00	No color	NULL	NULL	NULL	NULL	Purchased	Not Manufactured
325	Decal 1	DC-8732	no subcat	no category	0.00	No color	NULL	NULL	NULL	NULL	Purchased	Manufactured
326	Decal 2	DC-8824	no subcat	no category	0.00	No color	NULL	NULL	NULL	NULL	Purchased	Manufactured

Query executed successfully. | DESKTOP-GHPLDUQ (14.0 RTM) | DESKTOP-GHPLDUQ\Ola (53) | AdventureWorks2017 | 00:00:00 | 504 rows

b)

Setting necessary constraints

```
ALTER TABLE DWA.DIMProduct
```

```
ALTER COLUMN ProductID INTEGER NOT NULL
```

```
ALTER TABLE DWA.DIMProduct
```

```
ADD CONSTRAINT PK_ProductionProductID PRIMARY KEY (ProductID)
```

```
ALTER TABLE DWA.DIMProduct
```

```
ADD CONSTRAINT CK_Product_Class
```

```
CHECK (UPPER([Class])='H' OR UPPER([Class])='M' OR UPPER([Class])='L' OR [Class] IS NULL)
```

```
ALTER TABLE DWA.DIMProduct
```

```
ADD CONSTRAINT CK_Product_ProductLine
```

```
CHECK (UPPER([ProductLine])='R' OR UPPER([ProductLine])='M' OR  
UPPER([ProductLine])='T' OR UPPER([ProductLine])='S' OR [ProductLine] IS NULL)
```

```
ALTER TABLE DWA.DIMProduct
```

```
ADD CONSTRAINT CK_Product_ListPrice
```

```
CHECK ([ListPrice]>=(0.00))
```

```
ALTER TABLE DWA.DIMProduct
```

```
ADD CONSTRAINT CK_Product_Weight
```

```
CHECK ([Weight]>(0.00))
```

c) Verifying correctness

```
SELECT COALESCE(p.Color, 'noColor') AS Color, COUNT(COALESCE(p.Color, '')) AS  
ColorsinAdventureWorks, COUNT(dp.Color) AS NoOfColors
```

```
FROM AdventureWorks2017.Production.Product p
```

```
INNER JOIN DWA.DIMProduct dp
```

```
ON dp.ProductID = p.ProductID
```

```
GROUP BY p.Color
```

	Color	ColorsinAdventureWorks	NoOfColors
1	noColor	248	248
2	Black	93	93
3	Blue	26	26
4	Grey	1	1
5	Multi	8	8
6	Red	38	38
7	Silver	43	43
8	Silver/Black	7	7
9	White	4	4
10	Yellow	36	36


```
SELECT COUNT(ProductID) AS dimproduct
```

```
FROM DWA.DIMProduct
```

```
SELECT COUNT(ProductID) AS advworks
```

```
FROM AdventureWorks2017.Production.Product
```

dimproduct	
1	504

advworks	
1	504

Task 2.4

```
SELECT soh.SalesOrderID AS TransactionID, sod.SalesOrderDetailID AS  
TransactionDetailID, convert(varchar, OrderDate, 23) as OrderDate, convert(varchar,  
DueDate, 23) as DueDate, convert(varchar, ShipDate, 23) as ShipDate, sod.ProductID AS  
prodID, sod.UnitPrice AS UnitPrice, sod.OrderQty AS Quantity, LineTotal AS totalvalue,  
sod.UnitPriceDiscount AS UnitPriceDiscount, ( sod.UnitPriceDiscount*sod.OrderQty)AS  
DiscountInTotal
```

```
INTO dwaw.Salesinfo
```

```
FROM AdventureWorks2017.Sales.SalesOrderHeader soh
```

```
JOIN AdventureWorks2017.Sales.SalesOrderDetail sod
```

ON sod.SalesOrderID=soh.SalesOrderID

For this task we have prepared information about sales , particularly divided into information about orders and then about products. Our sales informations from the left is the orders ID number, then it focuses on all the products that were in that particular order number. That is why we have repeating productsID but we have decided that in order to include the unit price which is off an individual product, as well as orderID which is from each order, whilst showing the quantity and money of each product subtotal , that this is an optimal presentation of information and is more understandable .

TransactionID	TransactionDetailID	OrderDate	DueDate	ShipDate	prodID	UnitPrice	Quantity	totalvalue	UnitPriceDiscount	DiscountInTotal
43659	1	2011-05-31	2011-06-12	2011-06-07	776	2024.994	1	2024.994000	0.00	0.00
43659	2	2011-05-31	2011-06-12	2011-06-07	777	2024.994	3	6074.982000	0.00	0.00
43659	3	2011-05-31	2011-06-12	2011-06-07	778	2024.994	1	2024.994000	0.00	0.00
43659	4	2011-05-31	2011-06-12	2011-06-07	771	2039.994	1	2039.994000	0.00	0.00
43659	5	2011-05-31	2011-06-12	2011-06-07	772	2039.994	1	2039.994000	0.00	0.00
43659	6	2011-05-31	2011-06-12	2011-06-07	773	2039.994	2	4079.988000	0.00	0.00
43659	7	2011-05-31	2011-06-12	2011-06-07	774	2039.994	1	2039.994000	0.00	0.00
43659	8	2011-05-31	2011-06-12	2011-06-07	714	28.8404	3	86.521200	0.00	0.00
43659	9	2011-05-31	2011-06-12	2011-06-07	716	28.8404	1	28.840400	0.00	0.00
43659	10	2011-05-31	2011-06-12	2011-06-07	709	5.70	6	34.200000	0.00	0.00
43659	11	2011-05-31	2011-06-12	2011-06-07	712	5.1865	2	10.373000	0.00	0.00
43659	12	2011-05-31	2011-06-12	2011-06-07	711	20.1865	4	80.746000	0.00	0.00
43660	13	2011-05-31	2011-06-12	2011-06-07	762	419.4589	1	419.458900	0.00	0.00
43660	14	2011-05-31	2011-06-12	2011-06-07	758	874.794	1	874.794000	0.00	0.00
43661	15	2011-05-31	2011-06-12	2011-06-07	745	899.76	1	899.760000	0.00	0.00

Query executed successfully.

DESKTOP-GHPLDUQ (14.0 RTM) | DESKTOP-GHPLDUQ\Ola (56) | AdventureWorks2017 | 00:00:03 | 121317 rows

Constraints

```
ALTER TABLE DWA.Salesinfo
```

```
ADD CONSTRAINT PK_SalesOrderDetailID PRIMARY KEY (TransactionDetailID);
```

```
ALTER TABLE DWA.Salesinfo
```

```
ADD FOREIGN KEY (prodID)
```

```
REFERENCES DWA.DIMProduct(ProductID);
```

```
ALTER TABLE DWA.Salesinfo
```

```
ALTER COLUMN TransactionDetailID INTEGER NOT NULL
```

```
ALTER TABLE DWA.Salesinfo
```

```
ALTER COLUMN ProdID INTEGER NOT NULL
```

```
ALTER TABLE DWAW.Salesinfo
```

```
ALTER COLUMN TransactionID INTEGER NOT NULL
```

```
ALTER TABLE DWAW.Salesinfo
```

```
WITH
```

```
CHECK
```

```
ADD CONSTRAINT [CK_SalesOrderDetail_UnitPrice]
```

```
CHECK (([UnitPrice]>=(0.00)))
```

```
ALTER TABLE DWAW.Salesinfo
```

```
WITH
```

```
CHECK
```

```
ADD CONSTRAINT [CK_SalesOrderDetail_OrderQty]
```

```
CHECK (([Quantity]>(0)))
```

```
ALTER TABLE DWAW.Salesinfo
```

```
WITH
```

```
CHECK
```

```
ADD CONSTRAINT [CK_SalesOrderDetail_UnitPriceDiscount]
```

```
CHECK (([UnitPriceDiscount]>=(0.00)))
```

Task 2.5

Yearly brake-down of sales amount

```
select Sum(Quantity) as SalesAmount, [year] as Years
```

```
from dwaw.Salesinfo s
```

```
join dwaw.dimDate d
```

```
ON s.OrderDate=d.[date]
```

`group by [year]`

Best selling product category in terms of quantity and value separately

There are two ways to interpret this task, one is to show the best category in terms of quantity sales /value and one to allow user to see for himself the best category by ordering the quantity /value of sales separately.

Highest Quantity

```
SELECT Category, sum (Quantity) as QuantityWhole
FROM dwaw.DIMProduct dp
inner JOIN dwaw.SalesInfo s
ON s.prodID=dp.ProductID
group by Category
ORDER by QuantityWhole desc
```

Category	QuantityWhole
Bikes	90268
Clothing	73670
Accessories	61932
Components	49044

User can see for himself which was the best sold category in terms of quantity

```
SELECT Category, SUM(Quantity) as MaxQuantity
FROM dwaw.DIMProduct dp
inner JOIN dwaw.SalesInfo s
ON s.prodID=dp.ProductID
GROUP BY Category
),B as
```

```
(
SELECT  Max(MaxQuantity) as mx
from A
)
```

```
SELECT Category, MaxQuantity
from A,B
where MaxQuantity=mx
```

Category	MaxQuantity
Bikes	90268

Here we provide user information about the category with highest quantity only

Highest sales value
with A as(

```
SELECT Category,SUM(totalvalue) as MaxTotalSalesValue
FROM dwaw.DIMProduct dp
inner JOIN dwaw.SalesInfo ss
ON prodID=ProductID
GROUP BY Category
```

```
),B as
```

```
(
SELECT MAX( MaxTotalSalesValue) as mx
from A
)
SELECT Category, MaxTotalSalesValue
from A,B
where MaxTotalSalesValue=mx
```

Category	MaxTotalSalesValue
Bikes	94651172.704731

Here we provide user information about the category with highest sales value only

```
SELECT Category, sum (totalvalue) as WholeSalesValue
FROM dwaw.DIMProduct dp
inner JOIN dwaw.SalesInfo s
ON s.prodID=dp.ProductID
group by Category
ORDER by WholeSalesValue desc
```

Category	WholeSalesValue
Bikes	94651172.704731
Components	11802593.286430
Clothing	2120542.524801
Accessories	1272072.883926

User can see for himself which was the best sold category in terms of sales value

Best selling color during weekends. We have an analogic approach to this subtask as the previous one. In this task this task best selling category turn out to be one, 'No color', we can either accept as a best selling category or choose to pick an existing real color, showing only best sold color in the weekends or best sold colors ordered. We have chosen to keep the no color value because it is a valuable information about sales.

```

with A as(

SELECT Color,SUM (Quantity) as quantity

FROM dwaw.DIMProduct dp

left JOIN dwaw.SalesInfo ss

ON prodID=ProductID

INNER JOIN dwaw.dimDate dd ON

dd.[date]=ss.OrderDate

WHERE( [day] = 6 or [day] =7)

GROUP BY Color

),B as

(

SELECT MAX(quantity)as mx

from A

)

```

```
SELECT Color, quantity
```

```
from A,B
```

```
where quantity=mx
```

Color	quantity
No color	2060

or let the user see for himself best selling color with the others.

```
SELECT Color,SUM (Quantity) as quantity
```

```
FROM dwaw.DIMProduct dp
```

```
inner JOIN dwaw.SalesInfo ss
```

```
ON prodID=ProductID
```

```
INNER JOIN dwaw.dimDate dd ON
```

```
dd.[date]=ss.OrderDate
```

```
WHERE( [day] = 6 or [day] =7)
```

```
group by Color
```

```
order by quantity desc
```


Color	quantity
No color	2060
Black	710
Yellow	355
Red	347
Multi	299
Blue	272
Silver	247
White	28

Average price and price discount depending on different category only for manufactured and sold at least once products, we have eliminated null values for no category , because the task was to depend average price and discount on categories, existing and no category is not an existing category

```
SELECT Category,AVG(ListPrice) AS AvgListPrice, avg(UnitPriceDiscount )as
AvgPriceDiscount
```

```
FROM dwaw.DIMProduct dp
```

```
JOIN dwaw.SalesInfo ss
```

```
on ss.prodID=dp.ProductID
```

```
WHERE Manufactured= Manufactured AND Purchased=Purchased
```

```
GROUP BY Category
```

Category	AvgListPrice	AvgPriceDiscount
Clothing	43,2621	0,002
Bikes	1672,3917	0,0065
Accessories	21,5881	0,0008
Components	433,8962	0,0001

Average „Time-to-Ship” (Difference in days between order date and shipment date) for each product subcategory. We have concluded that we need to use inner join in order to remove null values of the average difference, because the null values appear for those subcategories which did not have an order date and ship date yet.

```
SELECT Subcategory ,avg(DATEDIFF(DAY,OrderDate,ShipDate) )as AvgDifference
FROM dwaw.DIMProduct dp
JOIN dwaw.SalesInfo ss
on ss.prodID=dp.ProductID
group by Subcategory
```

Subcategory	AvgDifference
Bib-Shorts	7
Bike Racks	7
Bike Stands	7
Bottles and Cages	7
Bottom Brackets	7
Brakes	7
Caps	7
Chains	7
Cleaners	7
Cranksets	7
Derailleurs	7
Fenders	7
Forks	7
Gloves	7
Handlebars	7

Query executed successfully. | DESKTOP-GHPLDUQ (14.0 RTM) | DESKTOP-GHPLDUQ\Ola (60) | 245670 | 00:00:00 | 35 rows

Task 2.6

```
CREATE VIEW vProductSales AS
```

```
SELECT TransactionID,prodID,Name, Category, Subcategory ,Color,Purchased,
Manufactured, UnitPrice,( Quantity),totalvalue ,UnitPriceDiscount, DiscountInTotal,
OrderDate, ShipDate,[quarter],(DATEDIFF(DAY,OrderDate,ShipDate) )as DifferenceDays
```

```
FROM DWA.W.DIMProduct dp
```

```
left JOIN DWA.W.SalesInfo ss
```

```
ON dp.ProductID = ss.prodID
```

```
INNER JOIN dwaw.dimDate dd
```

ON dd.[date]=ss.OrderDate

TransactionID	prodID	Name	Category	Subcategory	Color	Purchased	Manufactured	UnitPrice	Quantity	totalvalue	UnitPriceDiscount
43659	776	Mountain-100 Black, 42	Bikes	Mountain Bikes	Black	Not purchased	Not Manufactured	2024,994	1	2024.994000	0.00
43659	777	Mountain-100 Black, 44	Bikes	Mountain Bikes	Black	Not purchased	Not Manufactured	2024,994	3	6074.982000	0.00
43659	778	Mountain-100 Black, 48	Bikes	Mountain Bikes	Black	Not purchased	Not Manufactured	2024,994	1	2024.994000	0.00
43659	771	Mountain-100 Silver, 38	Bikes	Mountain Bikes	Silver	Not purchased	Not Manufactured	2039,994	1	2039.994000	0.00
43659	772	Mountain-100 Silver, 42	Bikes	Mountain Bikes	Silver	Not purchased	Not Manufactured	2039,994	1	2039.994000	0.00
43659	773	Mountain-100 Silver, 44	Bikes	Mountain Bikes	Silver	Not purchased	Not Manufactured	2039,994	2	4079.988000	0.00
43659	774	Mountain-100 Silver, 48	Bikes	Mountain Bikes	Silver	Not purchased	Not Manufactured	2039,994	1	2039.994000	0.00
43659	714	Long-Sleeve Logo Jersey, M	Clothing	Jerseys	Multi	Purchased	Manufactured	28,8404	3	86.521200	0.00
43659	716	Long-Sleeve Logo Jersey, XL	Clothing	Jerseys	Multi	Purchased	Manufactured	28,8404	1	28.840400	0.00
43659	709	Mountain Bike Socks, M	Clothing	Socks	White	Not purchased	Manufactured	5,70	6	34.200000	0.00
43659	712	AWC Logo Cap	Clothing	Caps	Multi	Purchased	Manufactured	5,1865	2	10.373000	0.00
43659	711	Sport-100 Helmet, Blue	Accessories	Helmets	Blue	Purchased	Manufactured	20,1865	4	80.746000	0.00
43660	762	Road-650 Red, 44	Bikes	Road Bikes	Red	Purchased	Not Manufactured	419,4589	1	419.458900	0.00
43660	758	Road-450 Red, 52	Bikes	Road Bikes	Red	Not purchased	Not Manufactured	874,794	1	874.794000	0.00

Query executed successfully. | DESKTOP-GHPLDUQ (14.0 RTM) | DESKTOP-GHPLDUQ\Ola (60) | 245670 | 00:00:05 | 121317 rows

Here we have information about every product that was in each order id

Task 3.1

Individual Customer Perspective:

In this task we have picked a top 1 address of each customer, where the modified date was lastly modified, meaning that this is the most up to date address.

```
SELECT sc.CustomerID, pp.FirstName,pp.LastName,pp.MiddleName, pp.Title,
addr.AddressLine1, addr.PostalCode,addr.City, addr.Province, addr.Country,
addr.CountryCode, st.Name AS Region, st.[Group] AS 'Geographic Area Name'
```

```
INTO     dwaw.DIMCustomer
```

```
FROM     AdventureWorks2017.Sales.Customer sc
```

```
INNER JOIN AdventureWorks2017.Person.Person    pp
```

```
ON sc.PersonID = pp.BusinessEntityID
```

```
INNER JOIN AdventureWorks2017.Sales.SalesTerritory st
```

```
ON sc.TerritoryID = st.TerritoryID CROSS APPLY
```

(

```
SELECT TOP 1 Address.AddressLine1, Address.City, StateProvince.Name AS Province,  
CountryRegion.Name AS Country, CountryRegion.CountryRegionCode as CountryCode,  
Address.PostalCode
```

```
FROM AdventureWorks2017.Person.BusinessEntityAddress addr
```

```
INNER JOIN AdventureWorks2017.Person.Address
```

```
ON addr.AddressID = AdventureWorks2017.Person.Address.AddressID
```

```
INNER JOIN AdventureWorks2017.Person.StateProvince
```

```
ON AdventureWorks2017.Person.Address.StateProvinceID =  
AdventureWorks2017.Person.StateProvince.StateProvinceID
```

```
INNER JOIN AdventureWorks2017.Person.CountryRegion
```

```
ON AdventureWorks2017.Person.StateProvince.CountryRegionCode =  
AdventureWorks2017.Person.CountryRegion.CountryRegionCode
```

```
WHERE addr.BusinessEntityID = pp.BusinessEntityID
```

```
ORDER BY addr.ModifiedDate DESC
```

```
) AS addr
```

```
WHERE PersonType = 'IN';
```

```
ALTER TABLE dwaw.DIMCustomer
```

```
ALTER COLUMN CustomerID INTEGER NOT NULL;
```

```
ALTER TABLE dwaw.DIMCustomer
```

```
ADD PRIMARY KEY (CustomerID);
```

CHECKING if the number of customers with person type individual is equal to the one in adventureWorks2017

```
SELECT sc.CustomerID
```

```

FROM AdventureWorks2017.Sales.Customer sc

INNER JOIN AdventureWorks2017.Person.Person pp

ON sc.PersonID = pp.BusinessEntityID

WHERE PersonType = 'IN';

```

CustomerID
11000
11001
11002
11003
11004
11005
11006
11007
11008
11009
11010
11011
11012
11013
11014

Query executed successfully. | DESKTOP-GHPLDUQ (14.0 RTM) | DESKTOP-GHPLDUQ\Ola (51) | 245670 | 00:00:00 | 18484 rows

CustomerID	FirstName	LastName	MiddleName	Title	AddressLine1	PostalCode	City	Province	Country	CountryCode	Region
11377	David	Robinet	R.	Mr.	Pappelallee 6667	42651	Solingen	Nordrhein-Westfalen	Germany	DE	Germany
11913	Rebecca	Robinson	A.	Ms.	1861 Chinquapin Ct	3198	Seaford	Victoria	Australia	AU	Australia
11952	Dorothy	Robinson	B.	Ms.	4693 Mills Dr.	3220	Geelong	Victoria	Australia	AU	Australia
20164	Carol Ann	Rockne	F.	Ms.	1312 Skycrest Drive	LA1 1LN	Lancaster	England	United Kingdom	GB	United Kingdom
20211	Scott	Rodgers	M.	Mr.	9860 Brookview Drive	4169	East Brisbane	Queensland	Australia	AU	Australia
20562	Jim	Rodman	NULL	Mr.	2377 Joyce Dr	EM15	Esher-Molesey	England	United Kingdom	GB	United Kingdom
20668	Eric	Rothenberg	NULL	Mr.	9277 Country View Lane	94519	Concord	California	United States	US	Southwest
20813	Michael	Rothkugel	L.	Mr.	3552 Mildred Ln.	2065	St. Leonards	New South Wales	Australia	AU	Australia
21190	Pablo	Rovira Diez	NULL	Mr.	15, rue Descartes	4169	East Brisbane	Queensland	Australia	AU	Australia
21279	Linda	Rousey	R.	Ms.	5966 Sepulveda Ct.	3198	Seaford	Victoria	Australia	AU	Australia
21286	Luke	Roy	J.	Mr.	8625 Woodcrest Drive	90210	Beverly Hills	California	United States	US	Southwest
21403	Lisa	Roy	K.	Ms.	6030 Winter Drive	4700	Rockhampton	Queensland	Australia	AU	Australia
21867	Michael	Ruggiero	NULL	Mr.	5501, rue Lauriston	93400	Saint-Denis	Seine Saint Denis	France	FR	France
21945	Pearlie	Rusek	J.	Ms.	5154 Brannan Pl	3198	Seaford	Victoria	Australia	AU	Australia

Query executed successfully. | DESKTOP-GHPLDUQ (14.0 RTM) | DESKTOP-GHPLDUQ\Ola (51) | 245670 | 00:00:01 | 18484 rows

Task 3.2

Sale's Location Perspective:

```
SELECT st.TerritoryID AS LocationID, st.Name, pc.Name AS 'Country name', st.[Group] AS 'Geographic area name'
```

```
INTO DIMSalesLocation
```

```
FROM AdventureWorks2017.Sales.SalesTerritory st
```

```
INNER JOIN AdventureWorks2017.Person.CountryRegion pc
```

```
ON st.CountryRegionCode = pc.CountryRegionCode;
```

```
ALTER TABLE DIMSalesLocation
```

```
ADD PRIMARY KEY (LocationID);
```

```
ALTER TABLE DIMSalesLocation
```

```
ALTER COLUMN LocationID INTEGER NOT NULL
```

Using inner join because all places have a country.

LocationID	Name	Country name	Geographic area name
1	Northwest	United States	North America
2	Northeast	United States	North America
3	Central	United States	North America
4	Southwest	United States	North America
5	Southeast	United States	North America
6	Canada	Canada	North America
7	France	France	Europe
8	Germany	Germany	Europe
9	Australia	Australia	Pacific
10	United Kingdom	United Kingdom	Europe

Checks with adventureWorks

```
Select TerritoryID
```

```
FROM AdventureWorks2017.Sales.SalesTerritory
```

TerritoryID
10
2
6
1
9
5
7
4
8
3

Query executed successfully. | DESKTOP-GHPLDUQ (14.0 RTM) | DESKTOP-GHPLDUQ\Ola (62) | 245670 | 00:00:00 | 10 rows

Task 3.3

Sales Information:

```
SELECT sd.SalesOrderID AS TransactionID, SalesOrderDetailID AS TransactionDetailID,  
convert(date,DateID,23) as DateID,sh.TerritoryID AS LocationID,ProductID,  
sh.CustomerID,UnitPrice,OrderQty ,LineTotal, UnitPriceDiscount, (UnitPriceDiscount *  
OrderQty) as TotalDiscount
```

```
INTO dwaw.CustomerSales
```

```
FROM AdventureWorks2017.Sales.SalesOrderDetail sd
```

```
INNER JOIN AdventureWorks2017.Sales.SalesOrderHeader sh
```

```
ON sd.SalesOrderID =sh.SalesOrderID
```

```
INNER JOIN dwaw.dimDate
```

```
ON dwaw.dimDate.day = DAY(sh.OrderDate)
```

```
AND dwaw.DIMDate.month = MONTH(sh.OrderDate)
```

```
AND dwaw.DIMdate.year = YEAR(sh.OrderDate)

INNER JOIN AdventureWorks2017.Sales.Customer

ON sh.CustomerID = AdventureWorks2017.Sales.Customer.CustomerID

INNER JOIN AdventureWorks2017.Person.Person

ON AdventureWorks2017.Sales.Customer.PersonID =
AdventureWorks2017.Person.Person.BusinessEntityID

WHERE PersonType = 'IN'
```

```
ALTER TABLE dwaw.CustomerSales

ADD PRIMARY KEY (TransactionDetailID);
```

```
ALTER TABLE dwaw.CustomerSales

ALTER COLUMN TransactionDetailID INTEGER NOT NULL;
```

```
ALTER TABLE dwaw.CustomerSales

ALTER COLUMN TransactionID INTEGER NOT NULL;
```

```
ALTER TABLE dwaw.CustomerSales

ALTER COLUMN DateID DATE NOT NULL;
```

```
ALTER TABLE dwaw.CustomerSales

ALTER COLUMN CustomerID INTEGER NOT NULL;
```



```
ALTER TABLE dwaw.CustomerSales
```

```
ALTER COLUMN LocationID INTEGER NOT NULL;
```

```
ALTER TABLE dwaw.CustomerSales
```

```
ALTER COLUMN ProductID INTEGER NOT NULL;
```

```
ALTER TABLE dwaw.CustomerSales
```

```
ADD FOREIGN KEY (DateID) REFERENCES dwaw.DIMDate([date]);
```

```
ALTER TABLE dwaw.CustomerSales
```

```
ADD FOREIGN KEY (CustomerID) REFERENCES dwaw.DIMCustomer (CustomerID);
```

```
ALTER TABLE dwaw.CustomerSales
```

```
ADD FOREIGN KEY (LocationID) REFERENCES DIMSalesLocation(LocationID);
```

```
ALTER TABLE dwaw.CustomerSales
```

```
ADD FOREIGN KEY (ProductID) REFERENCES DWAW.DIMProduct(ProductID);
```

Task 4

Firstly we imported both csv files, secondly added a primary key in Content and foreign key in rating table. Afterwards we created a third table 'DIMProductRating2' and inserted the wanted columns. To not omit any ratings, we have used left join although in this case the result is the same as with inner join.

```
ALTER TABLE dwaw.DIMContinent
```

```
ADD PRIMARY KEY(ContinentID
```

```
ALTER TABLE dwaw.DIMProductionRating
```

```
ADD FOREIGN KEY (ContinentID) REFERENCES DWAW.DIMContinent(ContinentID);
```

```
SELECT Product, Rating City, ct.Continent, Gender=(case when Gender='M' then 'Male'  
else 'Female' end), DAY(Time) as d, MONTH(Time) as m, YEAR(TIME) AS Y
```

```
INTO dwaw.DIMProductRating2
```

```
FROM dwaw.DIMProductRating pr
```

```
left JOIN dwaw.DIMContinent ct
```

```
ON pr.ContinentID=ct.ContinentID
```

Product	Rating	City	Continent	Gender	d	m	Y
0	0.63	Barbados	North America	Female	9	9	2011
0	0	Bahrain	North America	Male	24	12	2011
0	0	Egypt	Africa	Male	14	4	2013
0	0.42	Chad	Asia	Female	11	7	2013
0	0	Ivory Coast	Africa	Female	13	12	2013
0	0.65	Puerto Rico	Europe	Male	20	4	2014
1	8.99	Uganda	Africa	Female	4	2	2014
1	9.4	Puerto Rico	Europe	Male	25	9	2012
1	9.56	Belgium	Australia & Oceania	Male	20	10	2012
1	8.85	Argentina	Africa	Female	4	1	2013
1	8.31	Hungary	Africa	Female	30	1	2012
1	9.77	East Timor	Europe	Male	28	2	2012
2	0.62	East Timor	Europe	Female	3	5	2011
2	0	China	Africa	Female	31	8	2012
2	0.7	Namibia	Europe	Male	19	1	2014

Query executed successfully.

DESKTOP-GHPLDUQ (14.0 RTM) | DESKTOP-GHPLDUQ\Ola (53) | 245670 | 00:00:00 | 3012 rows

Checking if the number of products rated is the same

`SELECT * FROM dwaw.DIMProductRating`

`order by product`

Product	Rating	City	ContinentID	Gender	Time
0	0.628875643	Barbados	1	F	2011-09-09 00:00:00.000
0	0	Bahrain	1	M	2011-12-24 00:00:00.000
0	0	Egypt	5	M	2013-04-14 00:00:00.000
0	0.416797307	Chad	4	F	2013-07-11 00:00:00.000
0	0	Ivory Coast	5	Fem	2013-12-13 00:00:00.000
0	0.649026236	Puerto Rico	3	M	2014-04-20 00:00:00.000
1	8.991167234	Uganda	5	F	2014-02-04 00:00:00.000
1	9.402963467	Puerto Rico	3	M	2012-09-25 00:00:00.000
1	9.557938905	Belgium	6	M	2012-10-20 00:00:00.000
1	8.847004807	Argentina	5	F	2013-01-04 00:00:00.000
1	8.309040635	Hungary	5	F	2012-01-30 00:00:00.000
1	9.773240069	East Timor	3	M	2012-02-28 00:00:00.000
2	0.615649211	East Timor	3	Fem	2011-05-03 00:00:00.000
2	0	China	5	Men	2012-08-31 00:00:00.000
2	0.695557772	Namibia	2	M	2014-01-19 00:00:00.000

Query executed successfully.

DESKTOP-GHPLDUQ (14.0 RTM) | DESKTOP-GHPLDUQ\Ola (53) | 245670 | 00:00:00 | 3012 rows

For establishing best rated category we have used Power BI, uploaded data from Table `dwaw.DIMProductRating2` and merged it together with the view from previous tasks 'vProductSales' on productID. We have used left join to include all products that have been rated and only get their categories, created two pivot tables and charts. File will be zipped in a separate folder.

Task 5

In Power bi