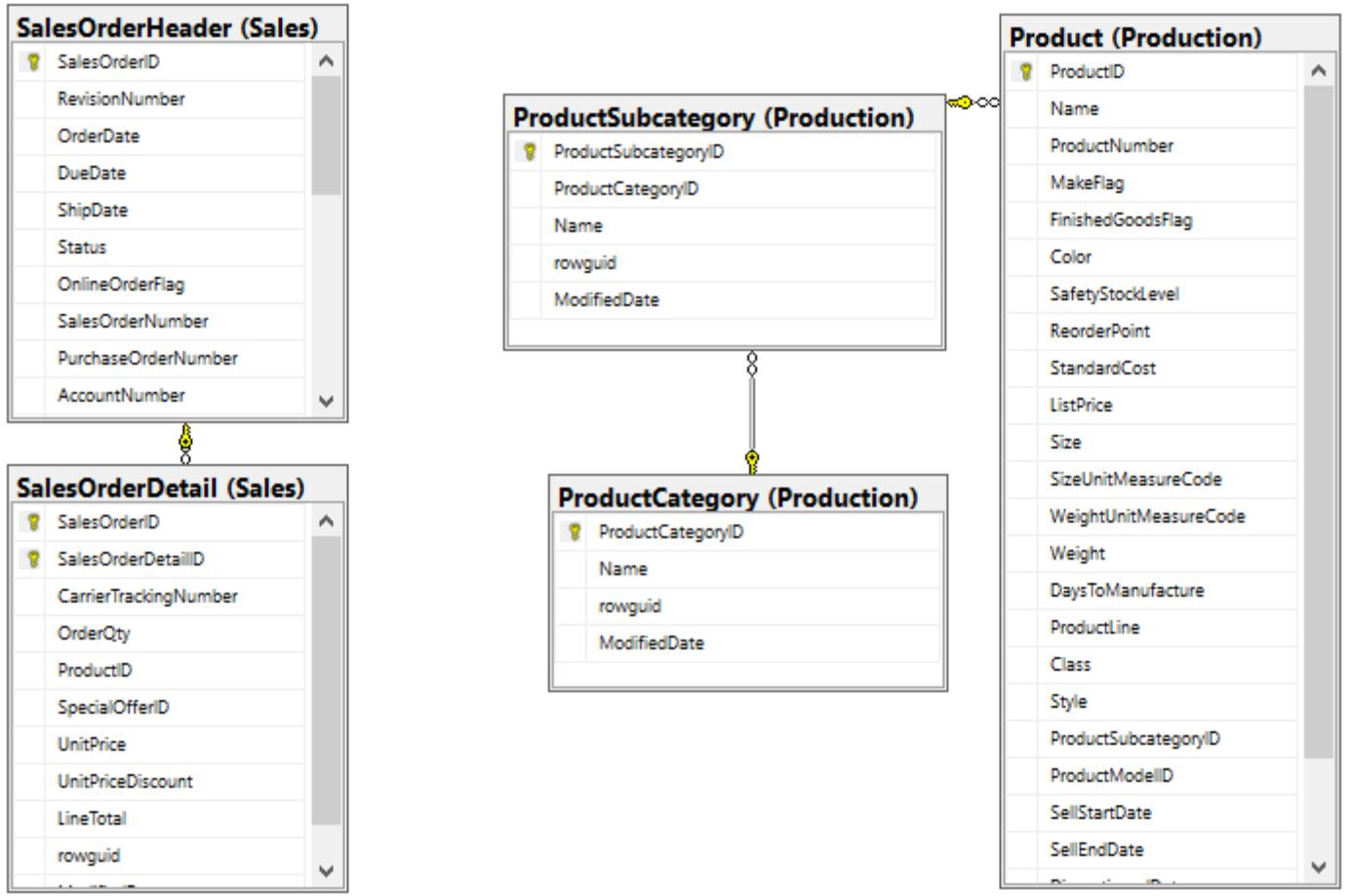
Tables from AdventureWorks2017 database used to the presentaton



Show products whose weight is known

```
SELECT ProductID, Name, Weight
FROM Production.Product
WHERE Weight IS NOT NULL
```

Results | Messages

	ProductID	Name	Weight
1	507	LL Mountain Rim	435.00
2	508	ML Mountain Rim	450.00
3	509	HL Mountain Rim	400.00
4	510	LL Road Rim	445.00
5	511	ML Road Rim	450.00

Remove nulls from product data

```
SELECT
Name,
COALESCE (Color, 'Not specified') as Color,
CASE
    WHEN Size IS NULL THEN 'Unknown'
    ELSE Size
END as Size
FROM Production.Product
```

	Name	Color	Size
206	Seat Tube	Not specified	Unknown
207	Top Tube	Not specified	Unknown
208	Tension Pulley	Not specified	Unknown
209	Rear Derailleur Cage	Silver	Unknown
210	HL Road Frame - Bl...	Black	58
211	HL Road Frame - R...	Red	58
212	Sport-100 Helmet, R...	Red	Unknown
213	Sport-100 Helmet, Bl...	Black	Unknown

Update the StandardCost of products in the 'Road Bikes' product subcategory. Increase the StandardCost by 15% for all products in this subcategory.

BEFORE UPDATE

```
SELECT pp.ProductID, pp.Name, pp.StandardCost, ps.Name
FROM Production.Product pp
JOIN Production.ProductSubcategory ps ON pp.ProductSubcategoryID = ps.ProductSubcategoryID
WHERE ps.Name = 'Road Bikes'
```

	ProductID	Name	StandardCost	Name
1	749	Road-150 Red, 62	2171,2942	Road Bikes
2	750	Road-150 Red, 44	2171,2942	Road Bikes
3	751	Road-150 Red, 48	2171,2942	Road Bikes

AFTER UPDATE

```
UPDATE Production.Product
SET StandardCost = StandardCost*1.15
WHERE ProductSubcategoryID IN (SELECT ProductSubcategoryID
                               FROM Production.ProductSubcategory
                               WHERE Name = 'Road Bikes')
```

	ProductID	Name	StandardCost	Name
1	749	Road-150 Red, 62	2496,9883	Road Bikes
2	750	Road-150 Red, 44	2496,9883	Road Bikes
3	751	Road-150 Red, 48	2496,9883	Road Bikes

List the total sales amount for each product category where the total sales amount is greater than $10,000. Include the product category name and the total sales amount in your result.

```
SELECT pc.Name as ProductCategory,
SUM(soh.TotalDue) as TotalSalesAmount
FROM Sales.SalesOrderHeader soh
LEFT JOIN Sales.SalesOrderDetail sod ON soh.SalesOrderID=sod.SalesOrderID
LEFT JOIN Production.Product pp ON sod.ProductID=pp.ProductID
LEFT JOIN Production.ProductSubcategory ps ON pp.ProductSubcategoryID=ps.ProductSubcategoryID
LEFT JOIN Production.ProductCategory pc ON ps.ProductCategoryID=pc.ProductCategoryID
GROUP BY pc.Name
HAVING SUM(soh.TotalDue)>10000
ORDER BY TotalSalesAmount desc
```

	ProductCategory	TotalSalesAmount
1	Bikes	1189845408.3579
2	Components	930569310.5143
3	Clothing	542468862.4396
4	Accessories	264086542.7296

HackerRank Challenges

Problem

Query the two cities in **STATION** with the shortest and longest CITY names, as well as their respective lengths (i.e.: number of characters in the name). If there is more than one smallest or largest city, choose the one that comes first when ordered alphabetically.

The **STATION** table is described as follows:

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

Solution

```
SELECT TOP 1 city,  LEN(CITY)
FROM STATION
ORDER BY LEN(CITY) asc, city;
```

```
SELECT TOP 1 city,  LEN(CITY)
FROM STATION
ORDER BY LEN(CITY) desc, city;
```

```
1  Amo 3
2  Marine On Saint Croix 21
```

We define an employee's total earnings to be their monthly $salary \times months$ worked, and the maximum total earnings to be the maximum total earnings for any employee in the **Employee** table. Write a query to find the maximum total earnings for all employees as well as the total number of employees who have maximum total earnings. Then print these values as 2 space-separated integers.

employee_id	name	months	salary
12228	Rose	15	1968
33645	Angela	1	3443
45692	Frank	17	1608
56118	Patrick	7	1345
59725	Lisa	11	2330
74197	Kimberly	16	4372
78454	Bonnie	8	1771
83565	Michael	6	2017
98607	Todd	5	3396
99989	Joe	9	3573

```
SELECT TOP 1 (SALARY * MONTHS), COUNT (*)
FROM EMPLOYEE
GROUP BY (SALARY * MONTHS)
ORDER BY (SALARY * MONTHS) DESC;
```

```
1   108064 7
```

Write a query to print the respective ***hacker_id*** and ***name*** of hackers who achieved full scores for *more than one* challenge. Order your output in descending order by the total number of challenges in which the hacker earned a full score. If more than one hacker received full scores in same number of challenges, then sort them by ascending *hacker_id*.

Hackers Table:

hacker_id	name
5580	Rose
8439	Angela
27205	Frank
52243	Patrick
52348	Lisa
57645	Kimberly
77726	Bonnie

Difficulty Table:

difficulty_level	score
1	20
2	30
3	40
4	60
5	80
6	100
7	120

Challenges Table:

challenge_id	hacker_id	difficulty_level
4810	77726	4
21089	27205	1
36566	5580	7
66730	52243	6
71055	52243	2

Submissions table:

submission_id	hacker_id	challenge_id	score
68628	77726	36566	30
65300	77726	21089	10
40326	52243	36566	77
8941	27205	4810	4
83554	77726	66730	30
43353	52243	66730	0
55385	52348	71055	20

```
SELECT h.hacker_id,  h.name
FROM hackers h
JOIN submissions s ON h.hacker_id=s.hacker_id
JOIN challenges c ON s.challenge_id=c.challenge_id
JOIN difficulty d ON c.difficulty_level=d.difficulty_level
WHERE s.score=d.score
GROUP BY h.hacker_id, h.name
HAVING COUNT(s.hacker_id)>1
ORDER BY COUNT(s.hacker_id) desc, h.hacker_id
```

```
1  27232 Phillip
2  28614 Willie
3  15719 Christina
4  43892 Roy
5  14246 David
```