INSTITUTO SUPERIOR TÉCNICO

Decision Support Systems
LEIC - Alameda
2010/2011

Homework #1

Due date: 3.Oct.2011

1 Data Description and Pre-processing

1. Suppose that a hospital tested the age and body fat data for 18 randomly selected adults with the following result:

Age	23	23	27	27	39	41	47	49	50
% Fat	9.5	26.5	7.8	17.8	31.4	25.9	27.4	27.2	31.2
Age	52	54	54	56	57	58	58	60	61
% Fat	34.6	42.5	28.8	33.4	30.2	34.1	32.9	41.2	35.7

(a) **(½ val.)** Calculate the mean, median and standard deviation for Age and % Fat. Indicate all relevant calculations.

Solution:
The mean of a set of values $\{x_1, \ldots, x_n\}$ is given by:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i.$$

In our case, this leads to:

$$\text{mean}_{\text{Age}} = \frac{1}{18} \times 836 = 46.4$$
$$\text{mean}_{\%\text{Fat}} = \frac{1}{18} \times 518.1 = 28.8$$

The standard deviation, in turn, is given by

$$\bar{s} = \left[\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2\right]^{1/2}$$

which, in our case, leads to

$$\text{std}_{\mathsf{Age}} = 12.8$$
$$\text{std}_{\%\mathsf{Fat}} = 9.0$$

Finally, the median of $\{x_1, \ldots, x_n\}$ is, roughly, the "middle" value when the elements of the set are ordered. In our case, this corresponds to:

$$\text{median}_{\mathsf{Age}} = 51$$
$$\text{median}_{\%\mathsf{Fat}} = 30.7$$

As a side note, the median can be computed approximately using the expression:

$$x_{\text{med}} \approx \frac{1}{2}(x_{\max} + x_{\min})$$

which, in our case, yields

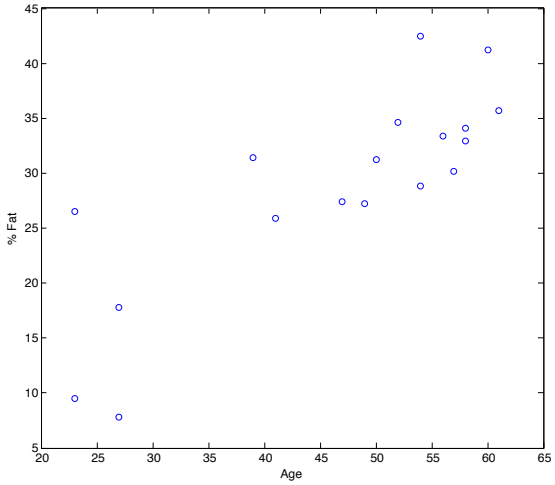
$$\text{median}_{\mathsf{Age}} \approx 42$$
$$\text{median}_{\%\mathsf{Fat}} \approx 25.15.$$

More refined approximations are possible by grouping the values in intervals, as seen in class.

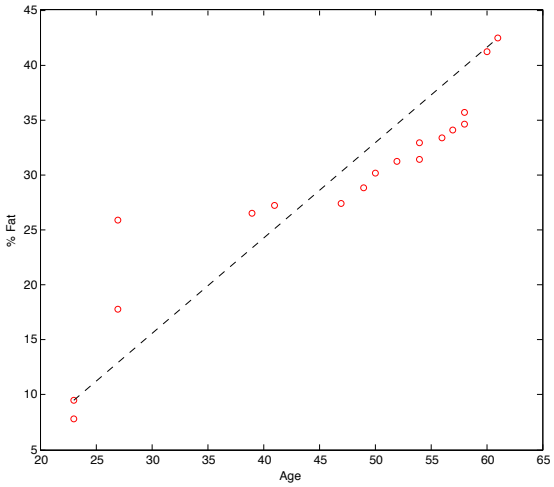
(b) **(½ val.)** Draw a *scatter plot* and a *q-q plot* based on the two variables.

Solution:

Recall that, in a scatter plot, we treat each pair of points $(\mathsf{Age}, \%\mathsf{Fat})$ as a pair of coordinates and plot all pairs as points in the plane. In our case, this leads to the plot:



As for the q-q plot, we must plot the quantiles of Age against those of % Fat. Since both sets have the same number of points, this corresponds to plotting the sorted values of Age against those of % Fat, leading to the plot:



(c) **(1 val.)** Normalize the two variables based on *z-score normalization*. Indicate all relevant calculations.

Solution:

Normalizing the two sets based on the z-score means that the data in each set must be translated and scaled according to the corresponding mean and standard deviation. In other words, each point x_i is transformed according to the expression:

$$x_i^N = \frac{x_i - \bar{x}}{\bar{s}}.$$

Since we had already computed the mean and standard deviation in Question (a), this leads to the following normalized sets:

Age^N	−1.82	−1.82	−1.51	−1.51	−0.58	−0.42	0.04	0.20	0.28
% Fat^N	−2.14	−0.25	−2.33	−1.22	0.29	−0.32	−0.15	−0.18	0.27

Age^N	0.43	0.59	0.59	0.74	0.82	0.90	0.90	1.06	1.13
% Fat^N	0.65	1.53	0.00	0.51	0.16	0.59	0.46	1.38	0.77

(d) **(1 val.)** Calculate the *correlation coefficient* (Pearson's product moment coefficient). Are these two variables positively or negatively correlated?

Solution:

The correlation coefficient between two sets $\{x_1, \ldots, x_n\}$ and $\{y_1, \ldots, y_n\}$ can be computed using the expression:

$$r_{X,Y} = \frac{1}{n\bar{s}_X\bar{s}_Y}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y}).$$

Since we already computed in Question (a) both the means and standard deviations for Age and % Fat, we have:

$$r_{\text{Age,\%Fat}} = 0.82.$$

We can conclude that both quantities are positively correlated, since $r_{\text{Age,\%Fat}} > 0$.

2. **(3 val.)** Let $\bar{x}_n$ be the mean of a set of n values $\{x_1, \ldots, x_n\}$. Suppose that you are given a new value, x_{n+1}, and must compute the updated mean, $\bar{x}_{n+1}$, of the set $\{x_1, \ldots, x_{n+1}\}$. Write down the expression for $\bar{x}_{n+1}$ *as a function of* $\bar{x}_n$, n and x_{n+1} (indicate all intermediate calculations).

Solution:

From the expression for $\bar{x}_{n+1}$ we get:

$$\begin{aligned}
\bar{x}_{n+1} &= \frac{1}{n+1}\sum_{i=1}^{n+1} x_i \\
&= \frac{1}{n+1}x_{n+1} + \frac{1}{n+1}\sum_{i=1}^{n} x_i \\
&= \frac{1}{n+1}x_{n+1} + \frac{n}{n+1}\frac{1}{n}\sum_{i=1}^{n} x_i \\
&= \frac{1}{n+1}x_{n+1} + \frac{n}{n+1}\bar{x}_n
\end{aligned}$$

which, after some shuffling, leads to

$$\bar{x}_{n+1} = \bar{x}_n + \frac{1}{n+1}\big(x_{n+1} - \bar{x}_n\big).$$

2 OLAP Queries

3. Suppose that a data warehouse consists of three dimensions: `Time`, `Doctor` and `Patient`, and the two measures `Count` and `Charge`, where `Charge` is the fee that a doctor charges a patient for a visit.

   (a) **(1 val.)** Starting with the cuboid [`Day`, `Doctor`, `Patient`], what specific OLAP operations should be performed in order to list the total fee collected by each doctor in 2004?

   Solution:

   The operations necessary to obtain the required information are:

   1. A *rollup* operation on the dimension `Time` to go from `Day` to `Year` (we implicitly assume that these two attributes are part of a hierarchy on the dimension `Time`).
   2. A *slice* operation on the dimension `Time` to conform with the restriction `Year`= 2004.
   3. A *rollup* on `Patient` from individual patients to `all`, in order to aggregate this particular dimension.

   (b) **(1 val.)** Write an SQL query to obtain the same list, assuming that the data are stored in a relational database with the scheme `Fee`(`Day`, `Month`, `Year`, `Doctor`, `Hospital`, `Patient`, `Count`, `Charge`). You can use the OLAP `CUBE` or `ROLLUP` operators with the syntax you saw in the lab.

   Solution:

   The required SQL query would be:

```
SELECT
   Doctor,
   SUM(Charge)
FROM
   Fee
WHERE
   Year=2004
GROUP BY
   Doctor
WITH ROLLUP
```

Note that the ROLLUP clause is not strictly necessary, but is included here for the sake of example.

4. **(2 val.)** Give an example of a pair of groupings with CUBE and ROLLUP that cannot be expressed by a single GROUP BY clause. In your example, you can use the JoBS database used in the lab.

Solution:

Resorting to the JoBS database, as suggested, the query

```
SELECT
   CASE WHEN GROUPING(E.EngineerName)=1
        THEN 'All Engineers' ELSE E.EngineerName END,
   CASE WHEN GROUPING(S.PartNumber)=1
        THEN 'All Parts' ELSE S.PartNumber END,
   SUM(S.UnitsHeld)
FROM
   EngineerStock S
INNER JOIN
   Engineers E
ON
   S.EngineerId = E.EngineerId
GROUP BY
   E.EngineerName,
   S.PartNumber
WITH ROLLUP
```

would require *three* GROUP BY clauses if the ROLLUP clause were not used:

```
SELECT
   E.EngineerName, S.PartNumber, SUM(S.UnitsHeld)
FROM
   EngineerStock S
INNER JOIN
   Engineers E
ON
   S.EngineerId = E.EngineerId
GROUP BY
   E.EngineerName,
   S.PartNumber
```

```
UNION
SELECT
   E.EngineerName, 'All arts', SUM(S.UnitsHeld)
FROM
   EngineerStock S
INNER JOIN
   Engineers E
ON
   S.EngineerId = E.EngineerId
GROUP BY
   E.EngineerName
UNION
SELECT
   'All Engineers', 'All Parts', SUM(S.UnitsHeld)
FROM
   EngineerStock S
INNER JOIN
   Engineers E
ON
   S.EngineerId = E.EngineerId
```

Similarly, the query

```
SELECT
   CASE WHEN GROUPING(E.EngineerName)=1
        THEN 'All Engineers' ELSE E.EngineerName END,
   CASE WHEN GROUPING(S.PartNumber)=1
        THEN 'All Parts' ELSE S.PartNumber END,
   SUM(S.UnitsHeld)
FROM
   EngineerStock S
INNER JOIN
   Engineers E
ON
   S.EngineerId = E.EngineerId
GROUP BY
   E.EngineerName,
   S.PartNumber
WITH CUBE
```

would require four GROUP BY queries to yield the same data.

2.1 Practical Questions (Using SQL Server 2008)

For the following questions, use the `AdventureWorksDW2008` database. This database is available in the lab. To this purpose, at the beginning of your code you should include the following SQL statement:

```
USE AdventureWorksDW2008;
GO
```

You can also use the Object Explorer in the MS SQL Server Management Studio to explore the tables in this database as well as the attributes in each table. For completeness, Fig. 1 includes a (simplified) representation of the relevant tables for this homework, where the attributes in italic correspond to primary keys.

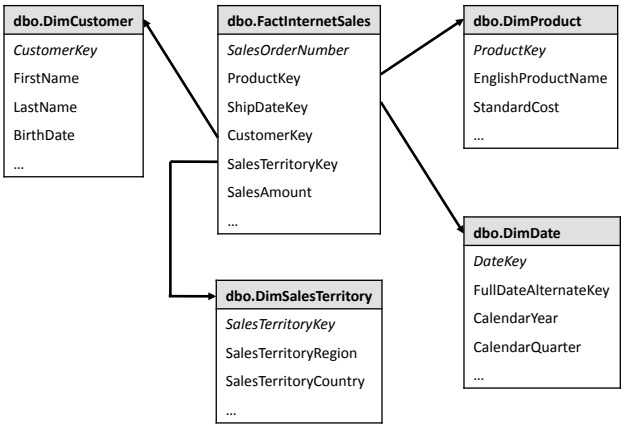


Figura 1: Simplified schema of the AdventureWorksDW2008 that includes only the relevant tables and attributes.

5. **(3 val.)** Write down the SQL query necessary to obtain the relation described in Fig. 2 from the table `dbo.FactInternetSales` (see Fig. 1). This relation describes, for each order in `dbo.FactInternetSales`, the first and last name of the corresponding customer, the region and country associated with the order, the name of the product in the order, its shipping date, and the total amount paid by the customer. In terms of the relations in Fig. 1,

(no name)
OrderNumber
FirstName
LastName
Region
Country
ProductName
ShipDate
Total

Figura 2: Table for question 5.

- The attribute `OrderNumber` in the new table corresponds to the attribute `SalesOrderNumber` in `dbo.FactInternetSales`;
- The attribute `Region` in the new table corresponds to the attribute `SalesTerritoryRegion` in the table `dbo.DimSalesTerritory`;
- The attribute `Country` in the new table corresponds to the attribute `SalesTerritoryCountry` in the table `dbo.DimSalesTerritory`;
- The attribute `ProductName` in the new table corresponds to the attribute `EnglishProductName` in `dbo.DimProduct`;
- The attribute `ShipDate` in the new table corresponds to the attribute `FullDateAlternateKey` in `dbo.DimDate`;

- The attribute `Total` in the new table corresponds to the attribute `SalesAmount` in the table `dbo.FactInternetSales`;

In your query you should use adequate JOIN operations.

Solution:

The SQL query would be:

```
SELECT
   F.SalesOrderNumber AS OrderNumber,
   C.FirstName,
   C.LastName,
   T.SalesTerritoryRegion AS Region,
   T.SalesTerritoryCountry AS Country,
   P.EnglishProductName AS ProductName,
   D.FullDataAlternateKey AS ShipDate,
   F.SalesAmount AS Total
FROM
   dbo.FactInternetSales F
LEFT JOIN
   dbo.DimCustomer C
ON
   F.CustomerKey = C.CustomerKey
LEFT JOIN
   dbo.dimSalesTerritory T
ON
   F.SalesTerritoryKey = T.SalesTerritoryKey
LEFT JOIN
   dbo.DimProduct P
ON
   F.ProductKey = P.ProductKey
LEFT JOIN
   dbo.DimDate D
ON
   F.ShipDateKey = D.DateKey
```

6. From the table `dbo.FactInternetSales` (see Fig. 1),
   (a) **(2 val.)** Determine the total sales amount per calendar year. Write down the corresponding SQL query.

Solution:

The SQL query is:

```
SELECT
   D.CalendarYear AS Year,
   SUM(F.SalesAmount) AS Total
FROM
   dbo.FactInternetSales F
INNER JOIN
   dbo.DimDate D
ON
```

```
   F.ShipDateKey = D.DateKey
GROUP BY
   D.CalendarYear
```

The corresponding values are:

Year	Total
2004	10,158,562.38
2001	3,105,587.33
2002	6,576,978.98
2003	9,517,548.53

(b) **(2 val.)** With a single query, determine both the global and per year the total sales amount (Suggestion: Use a ROLLUP clause). Write down the corresponding SQL query.

Solution:

The SQL query is:

```
SELECT
   D.CalendarYear AS Year,
   SUM(F.SalesAmount) AS Total
FROM
   dbo.FactInternetSales F
INNER JOIN
   dbo.DimDate D
ON
   F.ShipDateKey = D.DateKey
GROUP BY
   D.CalendarYear
WITH ROLLUP
```

The corresponding values are:

Year	Total
2004	10,158,562.38
2001	3,105,587.33
2002	6,576,978.98
2003	9,517,548.53
All	29,358,677.22

(c) **(3 val.)** Using the CUBE clause, determine the total sales amount across the two dimensions: `Year`, corresponding to the `CalendarYear` attribute in table `dbo.DimDate`, and `Country`, corresponding to the `SalesTerritoryCountry` attribute in table `dbo.DimSalesTerritory`. Write down the adequate SQL query and express the results as a *cross-tabulation*.

Solution:

The SQL query is:

```
SELECT
   D.CalendarYear AS Year,
   T.SalesTerritoryCountry AS Country,
   SUM(F.SalesAmount) AS Total
FROM
   dbo.FactInternetSales F
INNER JOIN
   dbo.DimDate D
ON
   F.ShipDateKey = D.DateKey
INNER JOIN
   dbo.DimSalesTerritory T
ON
   F.SalesTerritoryKey = T.SalesTerritoryKey
GROUP BY
   D.CalendarYear,
   T.SalesTerritoryCountry
WITH CUBE
```

The corresponding cross-tabulation is:

	2001	2002	2003	2004	All
Australia	1,251,388.1	2,166,222.5	3,002,149.1	2,641,240.9	9,061,000.6
Canada	143,251.5	618,206.8	507,224.8	709,161.8	1,977,844.9
France	172,716.1	508,910.0	992,681.6	969,710.0	2,644,017.7
Germany	219,372.8	528,003.6	1,021,797.2	1,125,138.8	2,894,312.3
United Kingdom	280,855.7	583,463.6	1,271,712.8	1,255,680.0	3,391,712.2
United States	1,028,003.1	2,172,172.5	2,721,983.1	3,457,630.8	9,389,789.5
All	3,105,587.3	6,576,979.0	9,517,548.5	10,158,562.4	29,358,677.2