

# Introduction à R - partie 2

Aurélie Labbe

Avant de commencer, assurez-vous que vous avez bien téléchargé tous les fichiers nécessaires pour la séance (pdf, code **R** et données) et que tout est sauvé dans un répertoire de votre ordinateur. Assurez-vous aussi que le code `IntroR_part2.R` est bien ouvert dans l'éditeur de RStudio. Finalement, vous devez avoir assimilé les notions de la 1e partie de l'introduction à **R** avant de commencer cette séance.

Dans ce document, nous allons couvrir les thèmes suivants:

- Statistiques descriptives
- Sélections de données
- Graphiques de base

## Données utilisées pendant la séance

### Les données électricité

Nous allons illustrer les différentes commandes et concepts à l'aide d'un fichier de données contenant la consommation en électricité d'une maison entre janvier 1991 et décembre 2000. Chaque ligne du fichier représente un mois donné. Les données sont dans les fichiers `electricbill.txt` et `electricbill.csv`. Les variables fournies sont les suivantes:

- **NUM**: identifiant de l'observation
- **YEAR**: année
- **MONTH**: mois
- **BILL**: montant de la facture en dollars (incluant 5% de taxe de vente)
- **TEMP**: température moyenne (en degrés Fahrenheit)
- **HDD**: mesure quantifiant la quantité d'énergie nécessaire pour chauffer le bâtiment
- **CDD**: mesure quantifiant la quantité d'énergie nécessaire pour refroidir le bâtiment
- **SIZE**: nombre de personnes vivant dans la maison
- **METER**: présence d'un nouveau compteur électrique (1=oui, 0=non)
- **PUMP1**: présence d'une nouvelle pompe de chauffage (1=oui, 0=non)
- **PUMP2**: présence d'une nouvelle pompe de chauffage pour la seconde fois (1=oui, 0=non)

- RIDER TOTAL: charge totale (par kwh)
- CONSUMPTION: consommation (en kwh)

## Les données baby-boom

Vous aurez à pratiquer les notions vues en classe à l'aide du jeu de données `babyboom.txt` qui contient l'heure de naissance, le sexe et le poids à la naissance de 44 bébés nés dans une période de 24h à l'hôpital de Brisbane, en Australie. Les variables sont les suivantes:

- Time: heure de naissance
- Sex: sexe du bébé (1 = fille, 2 = garçon)
- Weight: poids à la naissance en grammes
- Time.midnight: nombre de minutes après minuit passées au moment de la naissance

## Statistiques descriptives

Voici ci-dessous une liste de fonctions pré-définies dans **R** qui sont applicables pour des vecteurs (ex: colonnes de jeu de données):

```
# Lecture des données
donnees <- read.table("Data/electricbill.txt", header = TRUE) |>
  subset(select = c("BILL", "YEAR", "MONTH", "CONSUMPTION"))
BILL <- donnees$BILL
# Quelques opérations usuelles sur les vecteurs
length(BILL) # nombre d'entrées de l'objet
```

```
[1] 120
```

```
sum(BILL) # somme des éléments
```

```
[1] 12112.63
```

```
sum(BILL^2) # somme des éléments au carré
```

```
[1] 1539580
```

```
mean(BILL) # moyenne
```

```
[1] 100.9386
```

```
var(BILL) # variance
```

```
[1] 2663.434
```

```
sd(BILL) # écart-type (racine carrée de la variance)
```

```
[1] 51.60847
```

```
isTRUE(all.equal(sqrt(var(BILL)), sd(BILL)))
```

```
[1] TRUE
```

```
summary(BILL) # résumé
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
0.00	69.67	86.16	100.94	127.64	256.90

```
quantile(BILL, probs = 0.25) # 1e quartile
```

```
25%  
69.675
```

```
table(donnees$YEAR) # décompte par modalité
```

1991	1992	1993	1994	1995	1996	1997	1998	1999	2000
12	11	13	12	12	12	12	12	12	12

On peut aussi appliquer ces fonction à la table de données

```
summary(donnees)
```

BILL		YEAR	MONTH	CONSUMPTION		
Min.	: 0.00	Min.	:1991	Length:120	Min.	: 264
1st Qu.:	69.67	1st Qu.:	1993	Class :character	1st Qu.:	1796
Median :	86.16	Median :	1996	Mode :character	Median :	2602
Mean :	100.94	Mean :	1996		Mean :	3372
3rd Qu.:	127.64	3rd Qu.:	1998		3rd Qu.:	4500
Max.	:256.90	Max.	:2000		Max.	:13784

Notez que la documentation de **R** est très bien faite. Pour obtenir de l'aide sur une fonction (ex: `mean`), tapez la commande `help(mean)` et le fichier d'aide s'affichera dans la fenêtre en bas à droite de RStudio.

Il existe aussi beaucoup de fonctions spécifiques qui sont dans des paquets **R** que vous devez avoir installé au préalable. Par exemple, la fonction `mvrnorm`, qui permet de générer des variables selon une loi normale multivariée, est disponible dans le paquet **MASS** qui ne fait pas partie des paquets de base. Une fois que vous avez installé ce paquet, elle va faire partie de façon permanente des paquets que vous avez à votre disposition. Cependant, vous devez charger ce paquet à chaque nouvelle session de travail, si vous en avez besoin. Voici un exemple:

```
# Spécifier les valeurs des arguments avec le signe =  
x <- MASS::mvrnorm(100, mu = 1, Sigma = 1)
```

## Extraction de données

Il est très fréquent en analyse de données de devoir extraire un sous-ensemble de données pour faire des analyses plus spécifiques (ex: on veut extraire seulement les femmes, qui ont un emploi depuis plus de 10 ans). Nous avons déjà vu comment accéder à une ligne / colonne spécifique d'un dataframe avec le symbole `[ , ]` et nous allons voir qu'il est possible de combiner ceci avec des conditions de type logique (`FALSE` / `TRUE`).

Voyons tout d'abord comment créer une variable de type logique, à partir d'une condition spécifique. Par exemple, dans l'exemple suivant, la variable `condition` prend la valeur `TRUE` si la condition `YEAR=1991` est vraie (attention, la condition = se code `==` en R):

```
YEAR <- donnees$YEAR  
(condition <- YEAR == 1991)
```

```
[1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE  
[13] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  
[25] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  
[37] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  
[49] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  
[61] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  
[73] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  
[85] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  
[97] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  
[109] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

On pourrait aussi utiliser d'autres opérateurs logiques:

```
condition <- YEAR > 1995 # plus grand
condition <- YEAR >= 1995 # plus grand ou égal
condition <- YEAR != 1992 # différent de...
```

On peut extraire des valeurs d'un vecteur selon une condition à l'aide des symboles d'extraction `[]` comme suit:

```
# Extrait de la variable YEAR seulement les éléments pour lesquelles la condition est vraie
YEAR[YEAR > 1995]
```

```
[1] 1996 1996 1996 1996 1996 1996 1996 1996 1996 1996 1996 1996 1997 1997 1997
[16] 1997 1997 1997 1997 1997 1997 1997 1997 1997 1997 1998 1998 1998 1998 1998
[31] 1998 1998 1998 1998 1998 1998 1999 1999 1999 1999 1999 1999 1999 1999 1999
[46] 1999 1999 1999 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 2000
```

```
# Extrait de la variable BILL seulement les éléments correspondant à l'année 1991
donnees$BILL[YEAR == 1991]
```

```
[1] 162.10 256.90 151.15 118.76 100.71 83.97 99.40 103.64 71.59 94.92
[11] 78.83 182.85
```

De façon similaire, on peut aussi extraire des valeurs d'une matrice ou `data.frame`, selon une certaine condition. Voici des exemples:

```
# Extrait des données les lignes correspondant à l'année 1991
donnees[YEAR==1991,]
```

	BILL	YEAR	MONTH	CONSUMPTION
1	162.10	1991	Jan	5600
2	256.90	1991	Feb	9463
3	151.15	1991	Mar	5154
4	118.76	1991	Apr	3576
5	100.71	1991	May	2894
6	83.97	1991	Jun	2257
7	99.40	1991	Jul	2826
8	103.64	1991	Aug	2986
9	71.59	1991	Sep	1774
10	94.92	1991	Oct	2713
11	78.83	1991	Nov	2092
12	182.85	1991	Dec	6109

```
# Extrait des données les observations avec BILL>200
donnees[BILL>200,]
```

	BILL	YEAR	MONTH	CONSUMPTION
2	256.90	1991	Feb	9463
26	222.99	1993	Feb	7864
27	214.45	1993	Mar	7526
61	218.67	1996	Jan	11644
62	256.02	1996	Feb	13784
63	222.21	1996	Mar	12544

On peut aussi combiner ensemble plusieurs opérations logiques à l'aide des opérateurs & (ET) ou bien | (OU). Voici quelques exemples:

```
# Dimensions d'une nouvelle base de données qui contient
# seulement les mois de janvier et de février
# dim(donnees[donnees$MONTH == 'Jan' | donnees$MONTH == 'Feb', ])
sousbd <- donnees[donnees$MONTH %in% c('Jan','Feb'), ]
dim(sousbd)
```

```
[1] 20 4
```

```
head(sousbd)
```

	BILL	YEAR	MONTH	CONSUMPTION
1	162.10	1991	Jan	5600
2	256.90	1991	Feb	9463
13	165.23	1992	Jan	5400
14	197.94	1992	Feb	6657
25	180.08	1993	Jan	6163
26	222.99	1993	Feb	7864

```
# Seulement les observations avec 100<BILL<200
head(donnees[(BILL > 100) & (BILL < 200), ], nrow = 5L)
```

	BILL	YEAR	MONTH	CONSUMPTION
1	162.10	1991	Jan	5600
3	151.15	1991	Mar	5154
4	118.76	1991	Apr	3576
5	100.71	1991	May	2894
8	103.64	1991	Aug	2986
12	182.85	1991	Dec	6109

### Exercice 0.1.

- Créer une sous-base de données en extrayant de **donnees** seulement les mois de décembre à février, et les observations pour lesquelles **BILL** est supérieur à 100.
- Pour ce jeu de données, calculer la moyenne, la valeur minimum et la valeur maximum de la variable **BILL**.
- À l'aide de la fonction **cor**, calculer la corrélation entre les variables **BILL** et **CONSUMPTION**.

Pour terminer, nous allons voir comment extraire des données basées sur une séquence de ID que l'on veut extraire (par ex., extraire les 10 premières lignes d'un jeu de données). Voici tout d'abord comment créer en **R** une séquence de chiffres (ID):

```
# Séquence d'entiers de 5 à 10  
5:10
```

```
[1] 5 6 7 8 9 10
```

```
# Séquence de 1 à 10 avec des sauts de 2  
seq(from = 1L, to = 10L, by = 2L)
```

```
[1] 1 3 5 7 9
```

```
# Sequence de 0 à 10, de longueur 11  
seq(0, 10, length = 11)
```

```
[1] 0 1 2 3 4 5 6 7 8 9 10
```

```
# par défaut, R assigne les valeurs aux arguments dans l'ordre d'apparation  
  
# Répétition du chiffre 1, 10 fois  
rep(1, length.out = 10L)
```

```
[1] 1 1 1 1 1 1 1 1 1 1
```

```
# Répétition de la séquence 1-2-3, cinq fois  
rep(1:3, 5)
```

```
[1] 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3
```

En utilisant la même logique que ce que nous avons vu jusqu'à présent, il est possible d'extraire des lignes d'un tableau de données d'un vecteur ou d'une matrice.

```
# Extrait de BILL les observation (1,3,6)
BILL[c(1,3,6)]
```

```
[1] 162.10 151.15 83.97
```

```
# Extrait les 10 premières observations de BILL
BILL[1:10]
```

```
[1] 162.10 256.90 151.15 118.76 100.71 83.97 99.40 103.64 71.59 94.92
```

```
# Extrait les 10 premières lignes de mydata
head(donnees, n = 10)
```

	BILL	YEAR	MONTH	CONSUMPTION
1	162.10	1991	Jan	5600
2	256.90	1991	Feb	9463
3	151.15	1991	Mar	5154
4	118.76	1991	Apr	3576
5	100.71	1991	May	2894
6	83.97	1991	Jun	2257
7	99.40	1991	Jul	2826
8	103.64	1991	Aug	2986
9	71.59	1991	Sep	1774
10	94.92	1991	Oct	2713

```
# Extrait une ligne sur deux de mydata2 pour les colonnes (1,2,4)
donnees[seq(1, nrow(donnees), by = 2), -3]
```

	BILL	YEAR	CONSUMPTION
1	162.10	1991	5600
3	151.15	1991	5154
5	100.71	1991	2894
7	99.40	1991	2826
9	71.59	1991	1774
11	78.83	1991	2092
13	165.23	1992	5400
15	146.80	1992	4692



17	112.40	1992	3631
19	72.98	1992	1905
21	70.33	1992	1800
23	75.54	1992	2040
25	180.08	1993	6163
27	214.45	1993	7526
29	99.34	1993	2754
31	134.99	1993	4294
33	90.79	1993	2573
35	72.79	1993	1926
37	140.00	1994	5494
39	183.06	1994	6699
41	82.27	1994	2525
43	91.38	1994	2635
45	70.00	1994	1782
47	56.47	1994	2156
49	129.06	1995	6980
51	174.80	1995	8751
53	94.23	1995	4055
55	89.06	1995	3798
57	107.30	1995	4754
59	74.67	1995	2735
61	218.67	1996	11644
63	222.21	1996	12544
65	107.82	1996	5280
67	70.31	1996	2898
69	56.05	1996	1992
71	54.01	1996	2105
73	155.99	1997	4746
75	106.51	1997	2966
77	75.95	1997	1861
79	58.20	1997	1106
81	57.78	1997	1166
83	52.86	1997	1021
85	100.43	1998	2691
87	66.77	1998	1504
89	37.03	1998	562
91	42.44	1998	698
93	42.87	1998	710
95	34.49	1998	521
97	62.50	1999	1365
99	45.36	1999	805
101	28.23	1999	384

103	27.96	1999	378
105	153.32	1999	1887
107	80.96	1999	2043
109	126.86	2000	3658
111	121.19	2000	3455
113	115.70	2000	3311
115	91.23	2000	2434
117	83.97	2000	1320
119	72.30	2000	1055

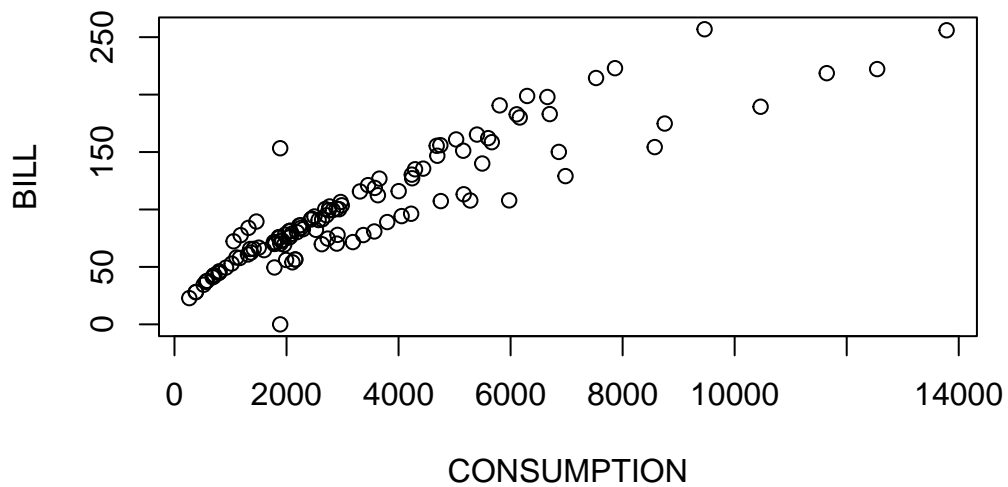
**Exercice 0.2.** Extraire du dataframe `donnees` les colonnes `YEAR` et `MONTH` pour les lignes paires.

## Graphiques

**R** peut produire une grande variété de graphiques de grande qualité et dans des formats variés (jpg, pdf, png, etc.). Il existe des paquets **R** telles que `lattice` et `ggplot2` qui permettent de créer des graphiques et figures de qualité exceptionnelle. La syntaxe de ces fonctions n'est pas forcément évidente pour un débutant dans **R**, et nous n'allons pas les détailler ici. La fonction de base `plot` permet d'effectuer des graphiques, mais la syntaxe est parfois réhibitoire. Le paquet `tinypplot` permet de simplifier les extensions sans introduire de dépendences additionnelles.

Voici le code qui utilise la fonction `plot` de base pour effectuer un simple nuage de points:

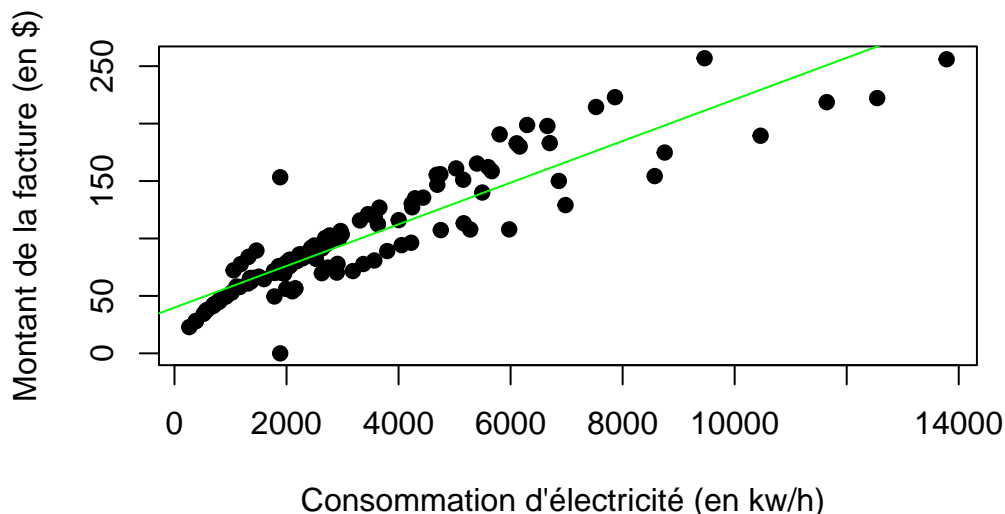
```
plot(BILL ~ CONSUMPTION, data = donnees)
```



Ajouter un titre et identifier les axes peut se faire facilement, de même que des couleurs ou des droites, ou des séparateurs verticaux ou horizontaux:

```
plot(BILL ~ CONSUMPTION, # formule sous forme y ~ x
     data = donnees,
     type = "p", # "p" pour point, "l" pour ligne, etc.
     xlab = "Consommation d'électricité (en kw/h) ",
     main = "Facture en fonction de la consommation d'électricité",
     ylab = "Montant de la facture (en $)",
     pch = 19) # type de point, cercle plein
abline(lm(BILL ~ CONSUMPTION, data = donnees), col = "green")
```

## Facture en fonction de la consommation d'électricité



```
# abline(h = 6000, col = "green") #ligne horizontale  
# abline(v = 100, col = "red") #ligne verticale
```

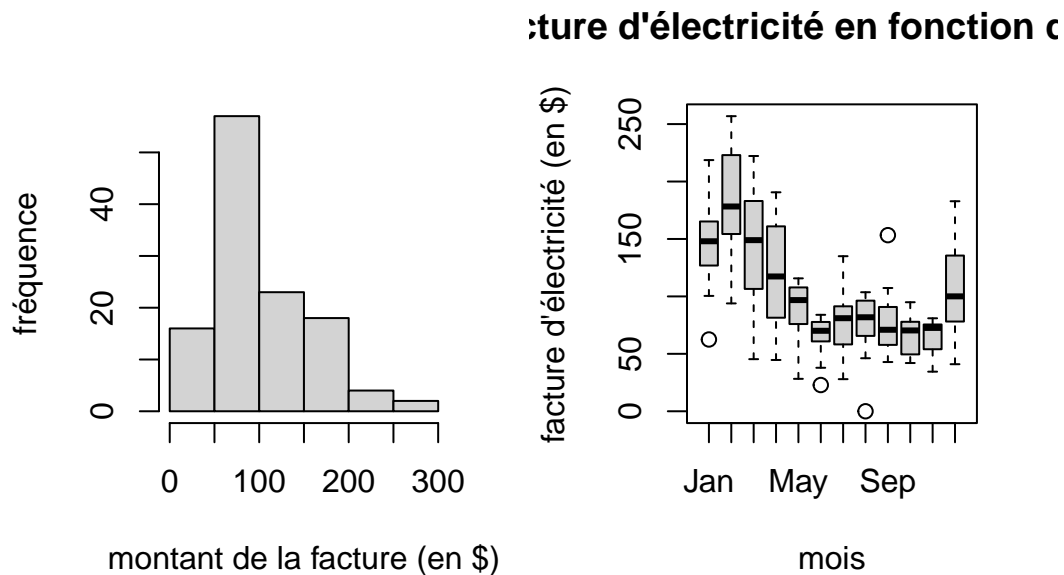
Plusieurs autres graphiques sont disponibles: ils dépendent du type des variables à disposition. Par exemple, on montre ci-dessous un histogramme et une boîte à moustaches. On peut tracer des boîtes à moustaches pour une seule variable, ou comme ici en fonction d'une variable catégorielle (on spécifie explicitement l'ordre des niveaux pour que les mois soient dans l'ordre chronologique plutôt qu'en ordre alphabétique). On change les options des paramètres graphiques (`par`) pour enlever la boîte autour du graphique et mettre les deux graphiques côte-à-côte.

```
par(mfrow = c(1, 2)) # configuration de la grille, 1 ligne et 2 colonnes<  
# Histogramme  
hist(BILL,  
      xlab = "montant de la facture (en $)",  
      ylab = "fréquence",  
      main = "")  
# Boîte à moustaches  
with(donnees,  
      boxplot(BILL ~ factor(MONTH,  
                             levels = substr(month.name, 1, 3)),  
              main = "Facture d'électricité en fonction du mois",
```

```

    ylab = "facture d'électricité (en $)",
    xlab = "mois")
)

```



```

par(mfrow = c(1, 1)) # rétablir les graphiques

```

Les graphes peuvent être sauves en format pdf, png, jpg (entre autres). Voici le code pour sauver une figure dans un fichier appelé “myplot.jpg”:

```

# créer un dossier pour stocker le graphique
dir.create(path = "figures", showWarnings = FALSE)
# enregistrer au format pdf au format 8 po par 4 po
pdf("figures/mongraphique.pdf", width = 8, height = 4)
plot(BILL ~ CONSUMPTION, data = donnees)
dev.off() # fermer la console graphique

```

pdf  
2

Si on exporte sous format PDF et que plusieurs graphiques sont créés, ils apparaîtront sur différentes pages.

### Exercice 0.3.

1. Faire un nuage de points de `BILL` versus `YEAR` séparément pour les mois de (1) janvier (2) mars (3) août et (4) octobre. Enregistrer ces 4 figures dans une seule page (2 lignes 2 colonnes) et le sauvegarder en format pdf.

### À vous de jouer

À l'aide de Quarto, répondez aux questions suivantes:

1. Calculer les statistiques descriptives de la variable `weight`.
2. Calculer les statistiques descriptives de la variable `weight` séparément pour les filles et les garçons
3. Produire deux boîtes à moustaches du poids à la naissance en fonction du sexe
4. Rapporter la proportion de filles qui ont un poids inférieur à 3 kg? Comparer avec la proportion chez les garçons.
5. Afficher le poids de toutes les filles qui sont nées au moins 2h après minuit.
6. Obtenir l'identifiant de la fille qui a le poids le plus faible.
7. Produire un nuage de points du poids en fonction de la variable `time.midnight`.