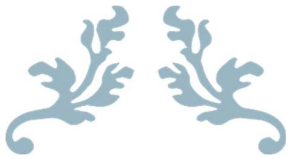


R-KIELEN PROJEKTI

ETL

JANNE BRAGGE

Sisällysluettelo

Datan luominen

Tässä luomme esimerkkimyyntejä sisältävän datan.

```
# Luo myyntidata
set.seed(123)  # Jotta tulokset ovat toistettavia


sales_data <- data.frame(
  Date = sample(seq(as.Date("2023-01-01"), as.Date("2023-12-31"), by = "day"), 100, replace = TRUE),
  Product = sample(c("Product A", "Product B", "Product C"), 100, replace = TRUE),
  Category = sample(c("Electronics", "Groceries", "Clothing"), 100, replace = TRUE),
  Quantity = sample(1:20, 100, replace = TRUE),
  Price = round(runif(100, 5, 100), 2)
)


# Tarkista data
head(sales_data)


# Tallenna data CSV-tiedostoon, jotta voimme käyttää sitä myöhemmin
write.csv(sales_data, "sales_data.csv", row.names = FALSE)
```

Tiedoston käyttö

Kun suoritat tämän koodin, se luo sales_data.csv-tiedoston, jonka voimme ladata ja käyttää seuraavissa vaiheissa.

Nyt kun tiedosto on luotu:

1. Varmista, että se on tallennettu samaan kansioon, josta suoritat R-koodia.
2. Voimme jatkaa **Extract**-vaiheeseen ja ladata tämän tiedoston käsittelyyn.

Vaihe 1: Extract - Datan lataus

Nyt ladataan juuri luotu CSV-tiedosto sales_data.csv R:ään ja tarkistetaan sen sisältö.

Koodi

Lataa data CSV-tiedostosta

sales_data <- read.csv("sales_data.csv", stringsAsFactors = FALSE)

Tarkista datan ensimmäiset rivit

head(sales_data)

```
> head(sales_data)
        Date   Product    Category Quantity Price
1 2023-06-28 Product C Electronics       16 13.71
2 2023-01-14 Product B    Clothing        2 84.05
3 2023-07-14 Product A    Clothing       15 31.30
4 2023-11-02 Product B Electronics        3 76.55
5 2023-04-28 Product B   Groceries        9 96.59
6 2023-10-26 Product B    Clothing        7 12.74
```

Tarkista datan rakenne

str(sales_data)

```
'data.frame':   100 obs. of  5 variables:
 $ Date    : chr  "2023-06-28" "2023-01-14" "2023-07-14" "2023-11-02" ...
 $ Product : chr  "Product C" "Product B" "Product A" "Product B" ...
 $ Category: chr  "Electronics" "Clothing" "Clothing" "Electronics" ...
 $ Quantity: int  16 2 15 3 9 7 9 4 2 12 ...
 $ Price   : num  13.7 84 31.3 76.5 96.6 ...
```

Tarkista, onko tiedostossa puuttuvia arvoja

sum(is.na(sales_data))

Selitys

1. **read.csv()**: Lataa CSV-tiedoston dataan. Käytämme stringsAsFactors = FALSE, jotta tekstimuotoiset sarakkeet pysyvät merkkijonoina, eivätkä muutu automaattisesti faktoreiksi.
2. **head()**: Näyttää datan ensimmäiset rivit, jotta voimme tarkistaa sen rakenteen.
3. **str()**: Näyttää datan rakenteen, kuten sarakkeiden nimet, tietotyypit ja muut tiedot.
4. **sum(is.na())**: Tarkistaa, onko datassa puuttuvia arvoja.

Odotettu tulos

- Näet myyntidatan, jossa on seuraavat sarakkeet:
    - **Date**: Myyntipäivämäärä
    - **Product**: Tuotteen nimi
    - **Category**: Tuotteen kategoria
    - **Quantity**: Myyty määrä
    - **Price**: Tuotteen yksikköhinta
- Varmistamme, että datassa ei ole puuttuvia arvoja.

Vaihe 2: Transform - Datan puhdistus ja rikastus

Nyt puhdistamme ja rikastamme dataa seuraavilla tavoilla:

1. Poistetaan mahdolliset puuttuvat arvot.
2. Muutetaan päivämäärät oikeaan muotoon.
3. Lasketaan kokonaishinta (määrä * yksikköhinta).
4. Luodaan uusi sarake, joka luokittelee tuotteet tuloluokkiin.

Koodi

```
# Lataa tarvittavat paketit

library(dplyr)

library(lubridate)


# 1. Poistetaan puuttuvat arvot

cleaned_data <- sales_data %>%

 drop_na()


# 2. Muutetaan päivämäärät oikeaan muotoon

cleaned_data <- cleaned_data %>%

 mutate(Date = as.Date(Date, format = "%Y-%m-%d"))


# 3. Lasketaan kokonaishinta
```

```
cleaned_data <- cleaned_data %>%
  mutate(TotalPrice = Quantity * Price)

# 4. Luodaan tuloluokkien sarake
cleaned_data <- cleaned_data %>%
  mutate(IncomeCategory = case_when(
    TotalPrice < 100 ~ "Low",
    TotalPrice >= 100 & TotalPrice < 500 ~ "Medium",
    TRUE ~ "High"
  ))

# Tarkista puhdistettu ja rikastettu data
head(cleaned_data)
```

	Date	Product	Category	Quantity	Price	TotalPrice	IncomeCategory
1	2023-06-28	Product C	Electronics	16	13.71	219.36	Medium
2	2023-01-14	Product B	Clothing	2	84.05	168.10	Medium
3	2023-07-14	Product A	Clothing	15	31.30	469.50	Medium
4	2023-11-02	Product B	Electronics	3	76.55	229.65	Medium
5	2023-04-28	Product B	Groceries	9	96.59	869.31	High
6	2023-10-26	Product B	Clothing	7	12.74	89.18	Low

Selitys

1. **drop_na()**: Poistaa rivit, joissa on puuttuvia arvoja.
2. **mutate()**:
    - Muuttaa Date-sarakkeen päivämäärämuotoon.
    - Laskee uuden sarakkeen TotalPrice, joka on myyntimäärä kerrottuna hinnalla.
    - Luo IncomeCategory-sarake, joka luokittelee tuotteet myynnin mukaan kolmeen tuloluokkaan:
        - **Low**: Kokonaishinta alle 100.
        - **Medium**: Kokonaishinta 100–500.
        - **High**: Kokonaishinta yli 500.

Odotettu tulos

- Data on nyt puhdistettu ja rikastettu seuraavilla sarakkeilla:

- **TotalPrice**: Lasketaan myynnin kokonaisarvo (määrä * hinta).
- **IncomeCategory**: Tuotteet jaoteltuna Low, Medium ja High -tuloluokkiin.

Vaihe 3: Yhteenvetojen luominen

Nyt luomme yhteenvetoja puhdistetusta ja rikastetusta datasta. Tavoitteena on:

1. Laskea kategorioiden kokonaissummat, keskihinnat ja kokonaismyyntimäärät.
2. Tarkastella, kuinka eri tuloluokat (Low, Medium, High) jakautuvat kategorioittain.

Koodi

```
# Lataa dplyr-paketti (jos ei jo ladattu)
library(dplyr)


# 1. Yhteenveto kategorioittain
category_summary <- cleaned_data %>%
 group_by(Category) %>%
 summarise(
  TotalSales = sum(TotalPrice),
  AveragePrice = mean(Price),
  TotalQuantity = sum(Quantity)
 )


# Näytä kategorioiden yhteenveto
print(category_summary)
```

```
# A tibble: 3 × 4
  Category    TotalSales AveragePrice TotalQuantity
  <chr>            <dbl>        <dbl>         <int>
1 Clothing        12563.         54.4           246
2 Electronics     29525.         53.3           556
3 Groceries       21322.         62.4           325
```

```
# 2. Tuloluokkien jakautuminen kategorioittain
income_summary <- cleaned_data %>%
 group_by(Category, IncomeCategory) %>%
 summarise(
  Count = n(),
  TotalSales = sum(TotalPrice)
 )
# Näytä tuloluokkien jakautuminen
print(income_summary)
```

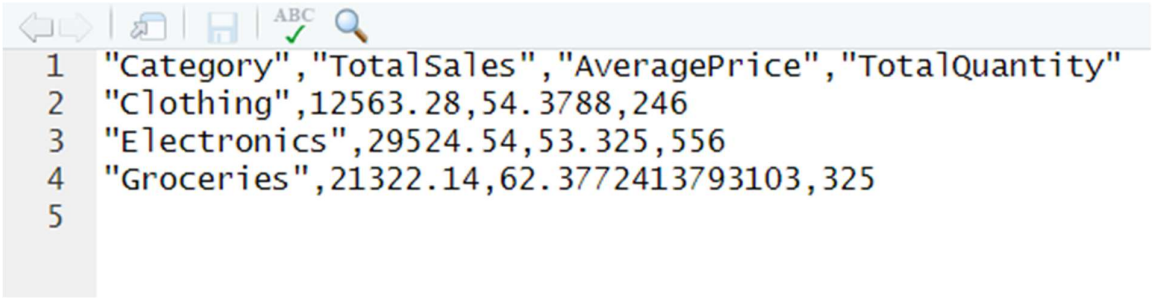
```
# A tibble: 9 × 4
# Groups:   Category [3]
  Category    IncomeCategory Count TotalSales
  <chr>       <chr>          <int>      <dbl>
1 Clothing    High               8      8304.
2 Clothing    Low                3       205.
3 Clothing    Medium            14      4054.
4 Electronics High              24     24428.
5 Electronics Low                5       306.
6 Electronics Medium            17      4791.
7 Groceries   High              19     19367.
8 Groceries   Low                5       341.
9 Groceries   Medium             5      1615.
>
```

Selitys

1. **group_by(Category)**: Ryhmittelee datan kategorian mukaan.
2. **summarise()**: Laskee yhteenvedot:
    - **TotalSales**: Kokonaissumma kategorian myynneistä.
    - **AveragePrice**: Keskihinta kategorian tuotteille.
    - **TotalQuantity**: Myytyjen tuotteiden kokonaismäärä.
3. **Tuloluokkien jakautuminen**:
    - **group_by(Category, IncomeCategory)**: Ryhmittelee datan sekä kategorian että tuloluokan mukaan.
    - **Count**: Laskee, kuinka monta myyntiä kuuluu kuhunkin ryhmään.
    - **TotalSales**: Laskee kokonaissumman kullekin tuloluokalle kategorioittain.

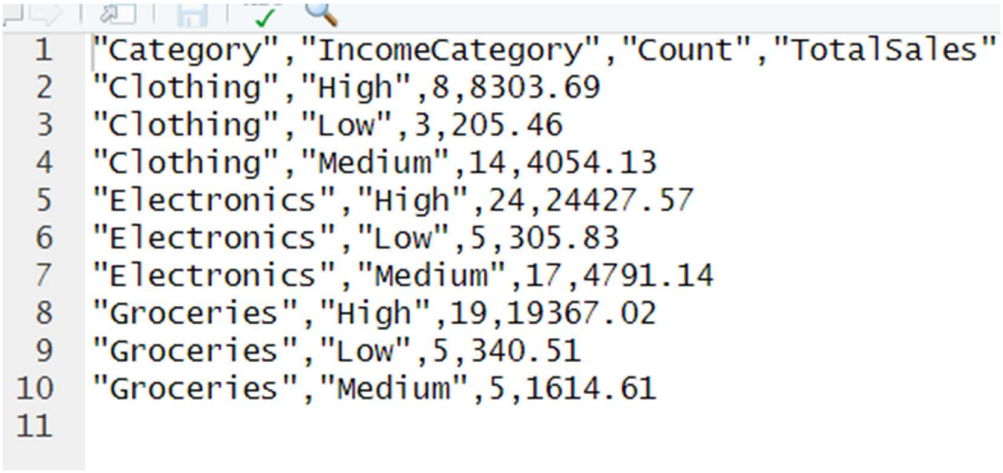
Odotettu tulos

Kategorioiden yhteenveto näyttää esimerkiksi:



```
"Category","TotalSales","AveragePrice","TotalQuantity"
"Clothing",12563.28,54.3788,246
"Electronics",29524.54,53.325,556
"Groceries",21322.14,62.3772413793103,325
```

Tuloluokkien jakautuminen kategorioittain näyttää esimerkiksi:



```
"Category","IncomeCategory","Count","TotalSales"
"Clothing","High",8,8303.69
"Clothing","Low",3,205.46
"Clothing","Medium",14,4054.13
"Electronics","High",24,24427.57
"Electronics","Low",5,305.83
"Electronics","Medium",17,4791.14
"Groceries","High",19,19367.02
"Groceries","Low",5,340.51
"Groceries","Medium",5,1614.61
```

Vaihe 4: Load - Datan tallennus

Tässä vaiheessa tallennamme:

1. **Puhdistetun ja rikastetun datan** uuteen CSV-tiedostoon.
2. **Kategorioiden yhteenvedon** ja **tuloluokkien jakautumisen** erillisiin tiedostoihin.

Koodi

```
# 1. Tallennetaan puhdistettu data
write.csv(cleaned_data, "cleaned_sales_data.csv", row.names = FALSE)

# 2. Tallennetaan kategorioiden yhteenveto
write.csv(category_summary, "category_summary.csv", row.names = FALSE)

# 3. Tallennetaan tuloluokkien jakautuminen kategorioittain
write.csv(income_summary, "income_summary.csv", row.names = FALSE)

# Vahvistusviesti
cat("Data on tallennettu seuraaviin tiedostoihin:\n",
  "- cleaned_sales_data.csv\n",
  "- category_summary.csv\n",
  "- income_summary.csv\n")
```

Selitys

1. **write.csv()**:
    - Tallentaa datan CSV-tiedostoon.
    - **row.names = FALSE** estää rivinumeroiden lisäämisen tiedostoon.
2. **cat()**: Tulostaa viestin, joka vahvistaa tallennuksen onnistuneen.

Odotetut tiedostot

1. **cleaned_sales_data.csv**: Puhdistettu ja rikastettu raakadata.
2. **category_summary.csv**: Kategorioiden kokonaissummat, keskihinnat ja myyntimäärät.

3. **income_summary.csv**: Tuloluokkien jakautuminen kategorioittain.

Vaihe 5: Datan tallentaminen tietokantaan

Voimme tallentaa puhdistetun ja käsitellyn datan SQLite-tietokantaan käyttäen R:n **DBI**- ja **RSQLite**-paketteja. Tämä prosessi luo tietokannan ja taulut, joissa data säilytetään.

Askel 1: Asenna ja lataa tarvittavat paketit

Jos paketteja ei ole vielä asennettu, asenna ne seuraavasti:

install.packages("DBI")

install.packages("RSQLite")

Lataa paketit:

library(DBI)

library(RSQLite)

Askel 2: Luo SQLite-tietokanta ja yhdistä

Luo tietokanta ja muodosta yhteys

con <- dbConnect(RSQLite::SQLite(), "sales_data.db")

Askel 3: Tallenna data tietokantaan

Tallennetaan puhdistettu data:

Tallenna cleaned_data-taulukko tietokantaan

dbWriteTable(con, "cleaned_sales_data", cleaned_data, overwrite = TRUE)

Tallennetaan yhteenvetotaulukot:

Tallenna category_summary-taulukko

dbWriteTable(con, "category_summary", category_summary, overwrite = TRUE)

Tallenna income_summary-taulukko

dbWriteTable(con, "income_summary", income_summary, overwrite = TRUE)

Askel 4: Tarkista tietokannan sisältö

Listaa kaikki taulut tietokannassa

```
dbListTables(con)


# Esikatsele taulua

dbReadTable(con, "cleaned_sales_data") %>% head()
```

	Date	Product	Category	Quantity	Price	TotalPrice	IncomeCategory
1	19536	Product C	Electronics	16	13.71	219.36	Medium
2	19371	Product B	Clothing	2	84.05	168.10	Medium
3	19552	Product A	Clothing	15	31.30	469.50	Medium
4	19663	Product B	Electronics	3	76.55	229.65	Medium
5	19475	Product B	Groceries	9	96.59	869.31	High
6	19656	Product B	Clothing	7	12.74	89.18	Low

Askel 5: Sulje yhteys

Kun kaikki on tehty, muista sulkea yhteys tietokantaan:

```
dbDisconnect(con)
```

Tietokantatiedoston sijainti

Tämä luo **sales_data.db**-tiedoston samaan hakemistoon, jossa suoritat R-skriptisi. SQLite-tietokanta sisältää kolme taulua:

1. **cleaned_sales_data**
2. **category_summary**
3. **income_summary**

Jatko-optiot

Voit käyttää tätä tietokantaa myöhemmin:

- **Datan noutamiseen ja analyysiin R:llä**:
- con <- dbConnect(RSQLite::SQLite(), "sales_data.db")
- data <- dbReadTable(con, "cleaned_sales_data")
- dbDisconnect(con)
- **Tietokannan käyttö SQL-komentojen avulla**:
- result <- dbGetQuery(con, "SELECT * FROM category_summary WHERE TotalSales > 5000")
- print(result)

Vaihe 6. Datavisualisointi: Myyntidatan analyysi

Tässä vaiheessa visualisoimme tietoja, jotka ovat syntyneet ETL-prosessin aikana. Käytämme **ggplot2**-kirjastoa, joka on tehokas ja monipuolinen työkalu datan visualisointiin.

Lataa ggplot2-paketti

Asenna ja lataa ggplot2, jos se ei ole jo käytössä:

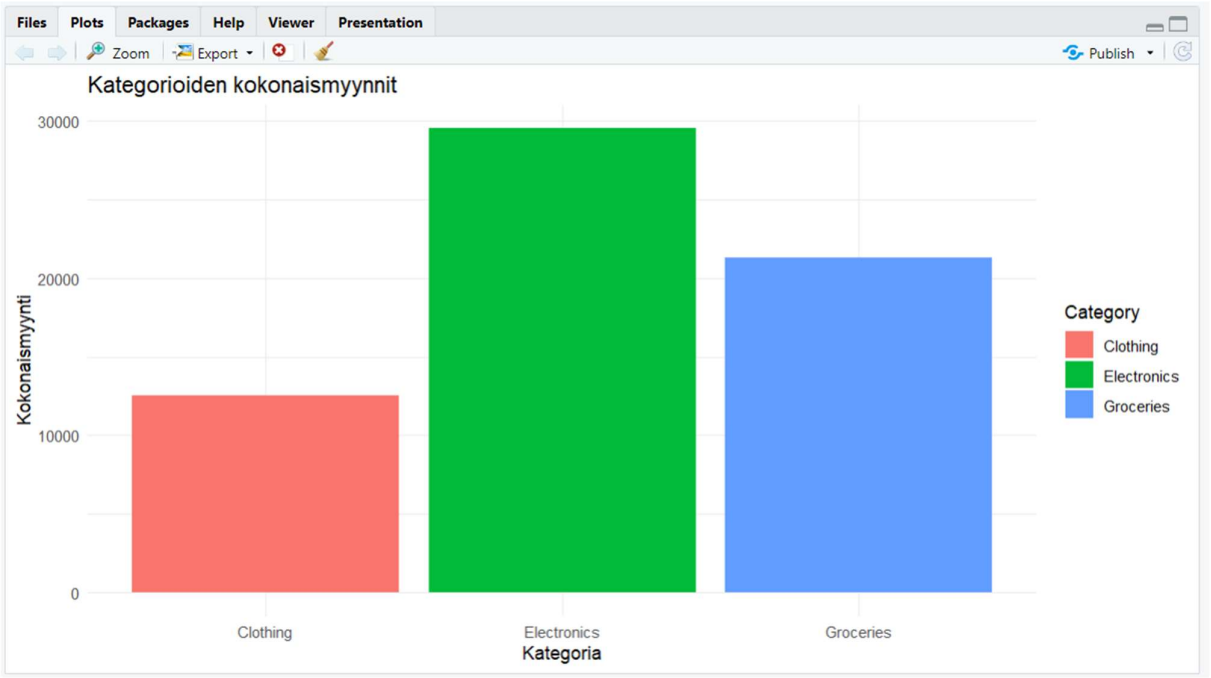
```
install.packages("ggplot2")
```

```
library(ggplot2)
```

Visualisointien luominen

1. Kokonaismyynnit kategorioittain

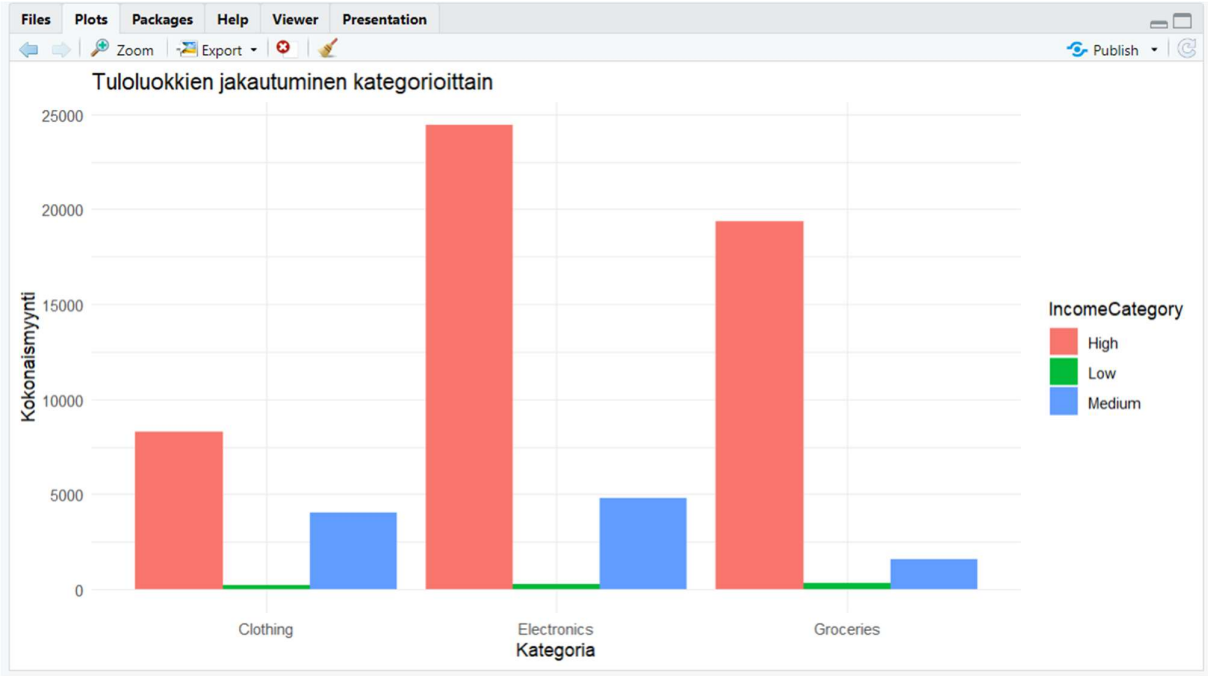
```
ggplot(category_summary, aes(x = Category, y = TotalSales, fill = Category)) +
  geom_bar(stat = "identity") +
  labs(title = "Kategorioiden kokonaismyynnit", x = "Kategoria", y = "Kokonaismyynti") +
  theme_minimal()
```



---

2. Tuloluokkien jakautuminen kategorioittain

```
ggplot(income_summary, aes(x = Category, y = TotalSales, fill = IncomeCategory)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Tuloluokkien jakautuminen kategorioittain",
       x = "Kategoria", y = "Kokonaismyynti") +
  theme_minimal()
```

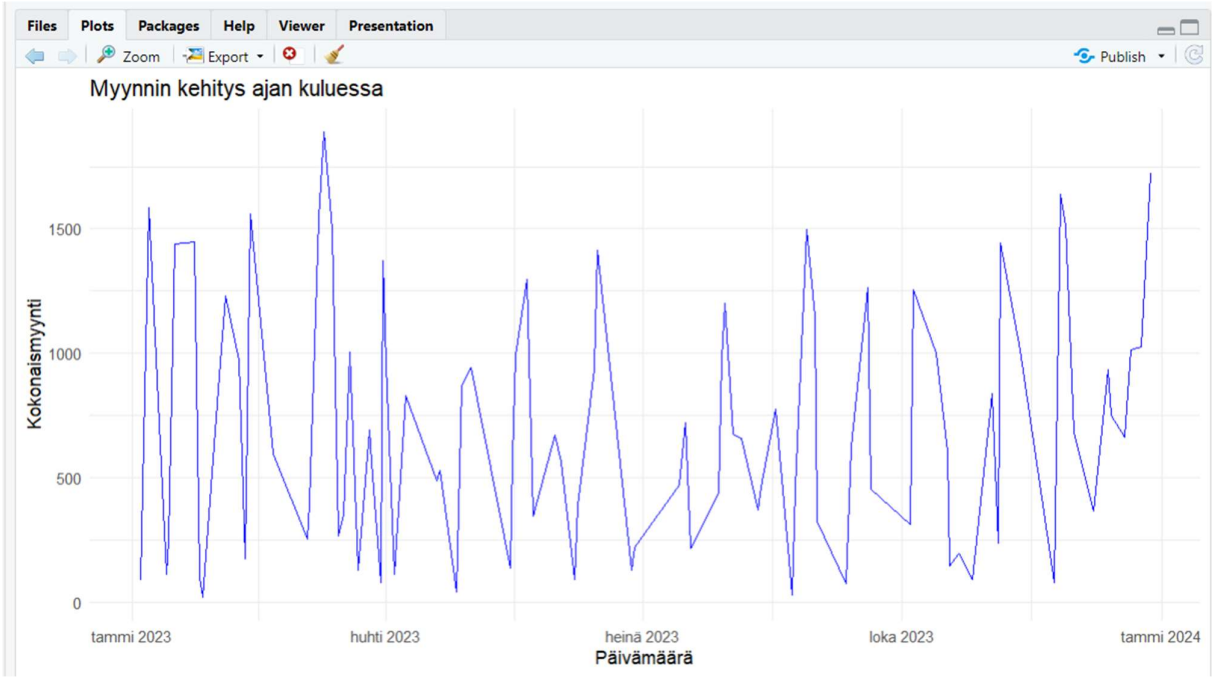


---

3. Myynnin kehitys ajan kuluessa

Jos haluat tarkastella myyntien kehitystä ajan kuluessa, voimme käyttää **puhdistettua dataa**:

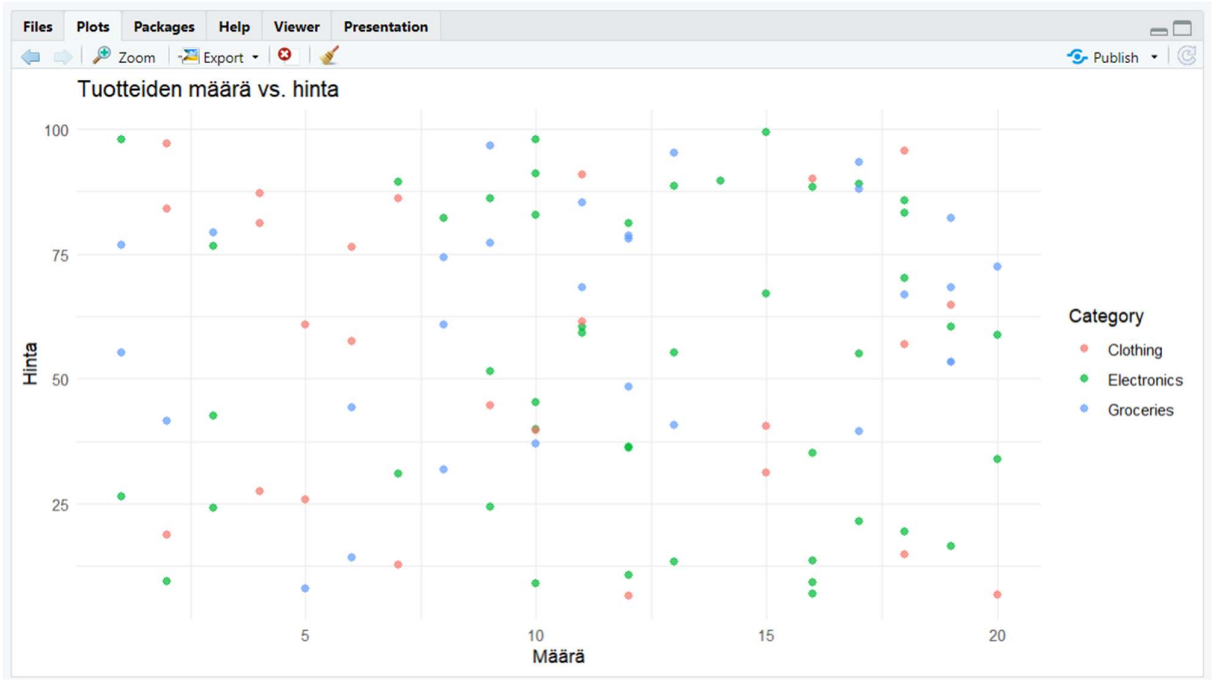
```
# Päivämääräkohtainen kokonaismyynti
daily_sales <- cleaned_data %>%
  group_by(Date) %>%
  summarise(DailyTotal = sum(TotalPrice))
# Viivakaavio myynnin kehityksestä
ggplot(daily_sales, aes(x = Date, y = DailyTotal)) +
  geom_line(color = "blue") +
  labs(title = "Myynnin kehitys ajan kuluessa", x = "Päivämäärä", y = "Kokonaismyynti") +
  theme_minimal()
```



---

4. Hajontakaavio tuotteiden määrän ja hinnan suhteesta

```
ggplot(cleaned_data, aes(x = Quantity, y = Price, color = Category)) +
 geom_point(size = 2, alpha = 0.7) +
 labs(title = "Tuotteiden määrä vs. hinta", x = "Määrä", y = "Hinta") +
 theme_minimal()
```



Lisävisualisoinnit

Historiallinen jakauma kokonaishinnan mukaan

```
ggplot(cleaned_data, aes(x = TotalPrice, fill = IncomeCategory)) +
 geom_histogram(bins = 20, alpha = 0.7) +
 labs(title = "Kokonaishintojen jakauma", x = "Kokonaishinta", y = "Frekvenssi") +
 theme_minimal()
```

Termien selitys

- **geom_bar(stat = "identity")**: Käytetään pylväsdiagrammien piirtämiseen, kun arvot ovat jo laskettuja.
- **geom_line()**: Piirtää viivakaavion.
- **geom_point()**: Luo hajontakaavion.
- **geom_histogram()**: Näyttää arvon jakauman histogrammina.
- **labs()**: Lisää otsikoita ja akseleiden selitteitä.
- **theme_minimal()**: Käyttää minimalistista teemaa, joka tekee kaaviosta modernin ja selkeän.

Odotetut tulokset

- Selkeitä ja visuaalisesti miellyttäviä kaavioita, jotka havainnollistavat datan piirteitä ja trendejä:
    - Kokonaismyyntien vertailu kategorioittain.
    - Tuloluokkien jakautuminen.
    - Myynnin kehitys ajan kuluessa.
    - Tuotteiden hinnan ja määrän välinen suhde.
