Employee Database R Analysis

Ian Brain

8/1/2022

Introduction

This R analysis comprises the Employee database utilized in the Employee SQL project and the Employee ER Diagram. This analysis uses a selection of queries from the Employee SQL project. This project differs in that it utilizes R to connect to the database and run queries. The project also uses R to create different visualizations.

Preparing the Project and Database

This project uses the tidyverse, GGally, and RMySQL package.

```
library(tidyverse)
library(GGally)
library(RMySQL)
```

To begin, R connects to the Employee database that has a host = localhost, port = 3306, username = root, and password = sqlpassword. This MySQL database is stored on a MySQL server on my computer.

```
mysqlemployee = dbConnect(RMySQL::MySQL(),
                          dbname='Employee',
                          host='localhost',
                          port=3306,
                          user='root',
                          password='sqlpassword')
```

A list of the tables in the the database are returned using dbListTables(). A list of the attributes in the employees table are then returned using dbListFields.

```
dbListTables(mysqlemployee)
```

```
## [1] "departments"  "dept_emp"     "dept_manager" "employees"    "salaries"
## [6] "titles"
```

```
dbListFields(mysqlemployee, 'employees')
```

```
## [1] "emp_no"     "birth_date" "first_name" "last_name"  "gender"
## [6] "hire_date"
```

Query Analysis

First, a query is executed to return the first name, last, name, and current title of each employee. Only the first five results are returned.

```
employeequery = dbSendQuery(mysqlemployee,
                      "select e.first_name AS first_name, e.last_name AS last_name, t.title AS title
                      from employees e
                      JOIN titles t
                         ON e.emp_no = t.emp_no
                      WHERE YEAR(t.to_date) = 9999")
eq.frame = fetch(employeequery, n=5)
print(eq.frame)
```

```
##   first_name last_name           title
## 1     Georgi   Facello Senior Engineer
## 2    Bezalel    Simmel           Staff
## 3      Parto   Bamford Senior Engineer
## 4  Chirstian   Koblick Senior Engineer
## 5    Kyoichi  Maliniak    Senior Staff
```

```
dbClearResult(employeequery)
```

```
## [1] TRUE
```

A query is then executed to fetch the average salary every year.

```
yearlysalary = dbSendQuery(mysqlemployee,
"
SELECT YEAR(from_date) AS 'year', AVG(salary) AS avg_salary
FROM salaries
GROUP BY YEAR(from_date)
ORDER BY YEAR(from_date)")
ys.frame = fetch(yearlysalary)
print(ys.frame[1:10, ])
```

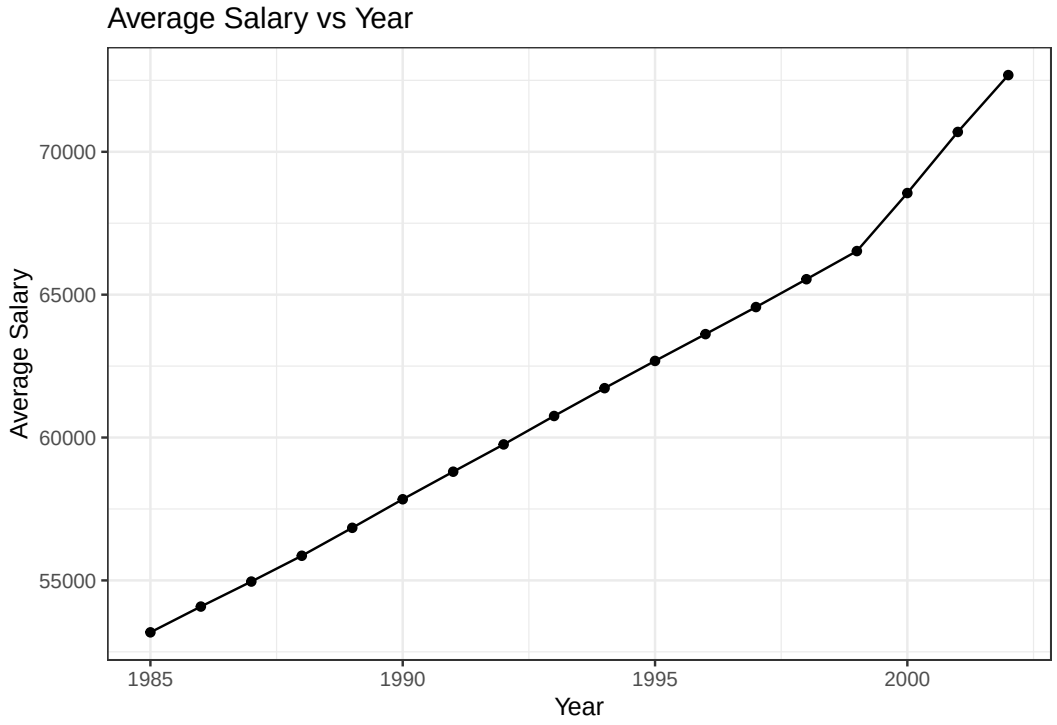
```
##    year avg_salary
## 1  1985   53182.36
## 2  1986   54084.78
## 3  1987   54959.63
## 4  1988   55862.45
## 5  1989   56840.67
## 6  1990   57839.46
## 7  1991   58803.87
## 8  1992   59758.74
## 9  1993   60753.66
## 10 1994   61727.76
```

```
dbClearResult(yearlysalary)
```

```
## [1] TRUE
```

A line plot is created for this query to visualize the result.

```
ggplot(ys.frame, aes(x = year, y = avg_salary)) +
  geom_line() +
  geom_point() +
  labs(x = "Year", y = "Average Salary") +
  ggtitle("Average Salary vs Year") +
  theme_bw()
```



Similarly, a query is executed to return the yearly average salary for each department. This query uses a CTE that returns the employee number, department, salary, and the respective year. This is then queried from to find the average salary each year, grouped by each department.

```
departmentsalary = dbSendQuery(mysqlemployee,
"
WITH dept_sal AS (
    SELECT e.emp_no, e.dept_no, s.salary, YEAR(s.to_date) AS year_end
    FROM salaries s
    JOIN dept_emp e
        ON s.emp_no = e.emp_no)
SELECT dept_name, year_end, avg_sal
FROM (
  SELECT dept_no, year_end, AVG(salary) AS avg_sal
  FROM dept_sal
  GROUP BY dept_no, year_end
  HAVING year_end > 1985
  ORDER BY dept_no, year_end) a
JOIN departments d
  ON a.dept_no = d.dept_no")
ds.frame = fetch(departmentsalary)
```

```
ds.frame$year_end <- replace(ds.frame$year_end, ds.frame$year_end == 9999, 2003)
print(ds.frame[1:10, ])
```

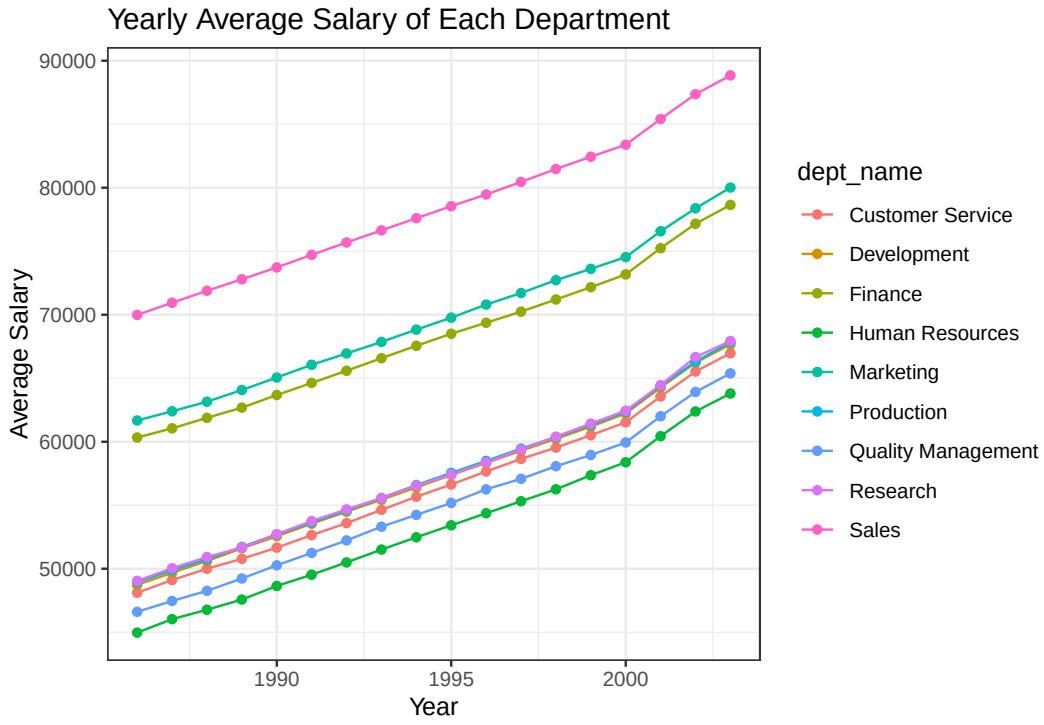
```
##           dept_name year_end  avg_sal
## 1  Customer Service     1986 48106.31
## 2  Customer Service     1987 49111.32
## 3  Customer Service     1988 50002.32
## 4  Customer Service     1989 50780.46
## 5  Customer Service     1990 51661.28
## 6  Customer Service     1991 52648.20
## 7  Customer Service     1992 53593.23
## 8  Customer Service     1993 54637.87
## 9  Customer Service     1994 55669.62
## 10 Customer Service     1995 56633.94
```

```
dbClearResult(departmentsalary)
```

```
## [1] TRUE
```

A line plot is created that contains each departments average salary every year.

```
ggplot(ds.frame, aes(x = year_end, y = avg_sal, group = dept_name)) +
  geom_line(aes(color = dept_name)) +
  geom_point(aes(color = dept_name)) +
  labs(x = "Year", y = "Average Salary") +
  ggtitle("Yearly Average Salary of Each Department") +
  theme_bw()
```



Next, a query is executed to return each salary in the current year along with the corresponding department name of the salary.

```
currentsalaries = dbSendQuery(mysqlemployee,
"
WITH department_names AS(
  SELECT emp_no, dept_name
  FROM dept_emp e
  JOIN departments d
    ON e.dept_no = d.dept_no),
emp_title AS (
  SELECT d.emp_no, dept_name, title
  FROM department_names d
  JOIN titles t
    ON t.emp_no = d.emp_no)
SELECT dept_name, salary, title
FROM salaries s
JOIN emp_title t
  ON s.emp_no = t.emp_no
WHERE YEAR(s.to_date) = 9999")
cs.frame = fetch(currentsalaries)
print(cs.frame[1:10, ])
```

```
##          dept_name salary           title
## 1      Development  88958 Senior Engineer
## 2            Sales  72527           Staff
## 3       Production  43311 Senior Engineer
## 4       Production  74057 Senior Engineer
## 5       Production  74057        Engineer
## 6  Human Resources  94692           Staff
## 7  Human Resources  94692    Senior Staff
## 8      Development  59755 Senior Engineer
## 9         Research  88070           Staff
## 10        Research  88070    Senior Staff
```

```
dbClearResult(currentsalaries)
```

```
## [1] TRUE
```

Numeric summaries are returned for the each department of the salary data using the group_by() function. The numeric summaries consist of the mean in addition to a 5 number summary.

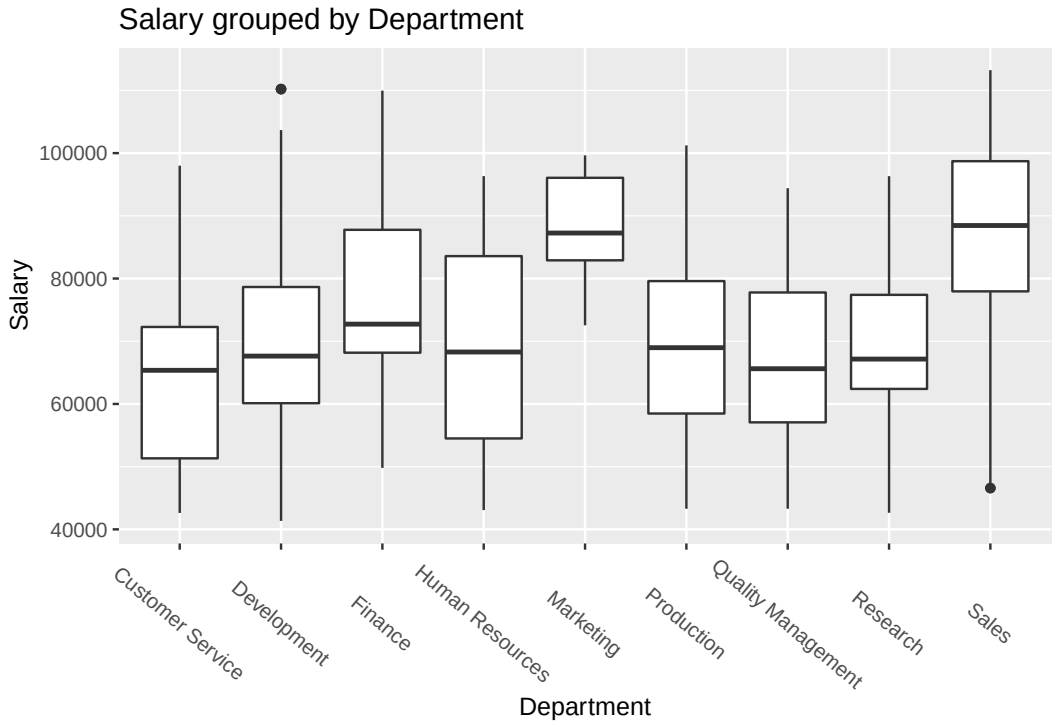
```
cs.frame %>%
  group_by(dept_name) %>%
  summarize(minimum = min(salary, na.rm =TRUE),
            Q1 = quantile(salary, probs = .25, na.rm = TRUE),
            mean = mean(salary, na.rm =TRUE),
            median = median(salary, na.rm =TRUE),
            Q3 = quantile(salary, probs = .75, na.rm = TRUE),
            maximum = max(salary, na.rm =TRUE))
```

```
## # A tibble: 9 x 7
##   dept_name          minimum     Q1   mean median     Q3 maximum
##   <chr>                <int>  <dbl>  <dbl>  <dbl>  <dbl>   <int>
## 1 Customer Service     42619 51326  64428. 65364  72275.   98003
```

```
## 2 Development          41348 60114  70013. 67630  78648.  110212
## 3 Finance              49802 68175  76489. 72729  87766.  109964
## 4 Human Resources      43073 54501. 70479. 68281  83574.   96333
## 5 Marketing            72542 82905  87755. 87254  96062    99651
## 6 Production           43283 58472  70117. 68974. 79594.  101239
## 7 Quality Management   43283 57066. 68167. 65617  77777    94409
## 8 Research             42646 62398  69604. 67156  77400.   96322
## 9 Sales                46583 77955  87554. 88454. 98711   113229
```

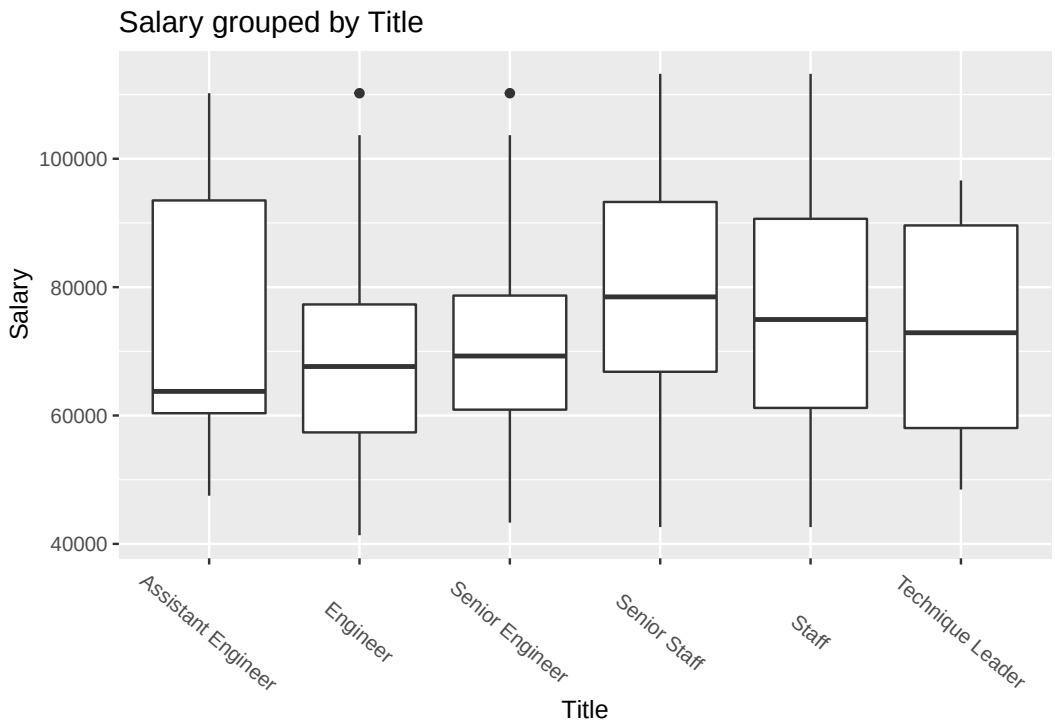
These 5 number summaries are then visualized using side by side box plots created for the values of salary that correspond with each department. The factor() function is used as department is a categorical variable.

```
ggplot(cs.frame, aes(x = factor(dept_name), y = salary)) +
  geom_boxplot() +
  theme(axis.text.x=element_text(angle = 320, vjust = 0.5)) +
  labs(x = "Department", y = "Salary") +
  ggtitle("Salary grouped by Department")
```



Similarly, side by side box plots created for the values of salary that correspond with each employee title. The factor() function is used as title is a categorical variable.

```
ggplot(cs.frame, aes(x = factor(title), y = salary)) +
  geom_boxplot() +
  theme(axis.text.x=element_text(angle = 320, vjust = 0.5)) +
  labs(x = "Title", y = "Salary") +
  ggtitle("Salary grouped by Title")
```



Finally, a query is executed to return the number of employees each department. This query is performed using a CTE that returns the employee number, manager number, current date, and department. The number of employees in each department are then counted by grouping by department.

```
dept_num_emp = dbSendQuery(mysqlemployee,
"
WITH man_emp AS (
    SELECT DISTINCT e.emp_no AS emp_no, m.emp_no AS mngr_no, m.to_date AS cur_date, e.dept_no AS dept_no
    FROM dept_emp e
    LEFT JOIN dept_manager m
        ON e.dept_no = m.dept_no
    WHERE m.to_date IN (SELECT MAX(to_date) FROM dept_manager GROUP BY dept_no)
    ORDER BY dept_no)
SELECT mngr_no, dept_name, num_emp
FROM (
  SELECT mngr_no, dept_no, COUNT(emp_no) AS num_emp
  FROM man_emp
  GROUP BY mngr_no, dept_no
  ORDER BY num_emp DESC) a
JOIN departments d
  ON a.dept_no = d.dept_no")
dne.frame = fetch(dept_num_emp)
print(dne.frame[1:10, ])
```

```
##    mngr_no          dept_name num_emp
## 1   111939   Customer Service   23580
## 2   110567        Development   85707
## 3   110114            Finance   17346
## 4   110228    Human Resources   17786
## 5   110039          Marketing   20211
```

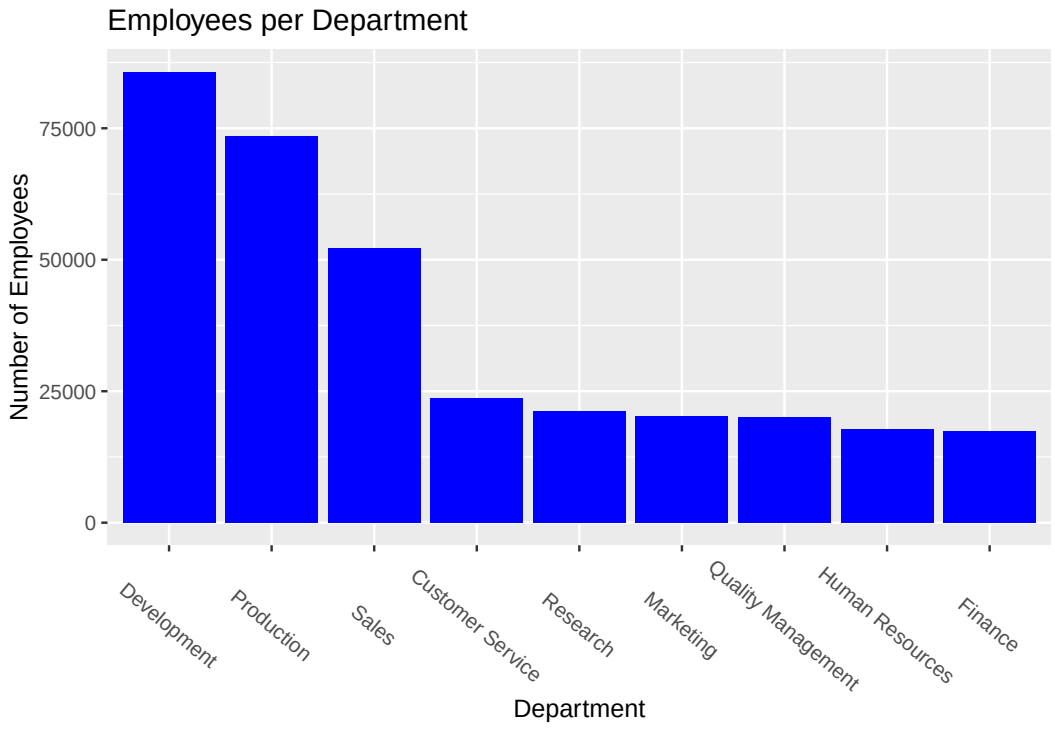
```
## 6   110420         Production   73485
## 7   110854 Quality Management   20117
## 8   111534           Research   21126
## 9   111133              Sales   52245
## NA      NA               <NA>      NA
```

```
dbClearResult(departmentsalary)
```

```
## [1] TRUE
```

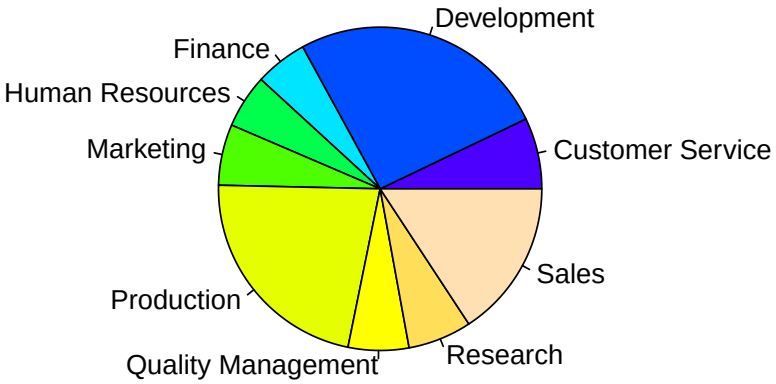
A bar chart is created to visualize the number of employees per department.

```
ggplot(dne.frame, aes(x = fct_rev(fct_reorder(dept_name, num_emp)), y = num_emp)) + geom_col(fill = "blu
  labs(x = "Department", y = "Number of Employees") +
  ggtitle("Employees per Department") +
  theme(axis.text.x=element_text(angle = 320, vjust = 0.5))
```



A pie chart is also created to visualize the number of employees per department.

```
pie(dne.frame$num_emp, dne.frame$dept_name, col = topo.colors(length(dne.frame$dept_name)))
```



R is then disconnected from the Employee database.

```
dbDisconnect(mysqlemployee)
```

```
## [1] TRUE
```

Sources

Resources used to understand dbplyr and RMySQL syntax:

https://www.projectpro.io/recipes/connect-mysql-r

https://www.linkedin.com/pulse/rmysql-tutorial-beginners-rambaksh-prajapati/

Employee database: https://relational.fit.cvut.cz/dataset/Employee