Challenge 9 - SQL

Name: Jeremy HOOPER

Date: 29 January 2024

Introduction

Challenge 9 SQL is divided into three parts to test our knowledge of three aspects of database design (modelling), creation (engineering), and analysis (reading). The three operations represent the first half of the foundation of most data management systems known by the acronym CRUD, or:

- **Create**: Adding new data or records to the database.
- **Read**: Retrieving or viewing existing data.
- **Update**: Modifying or changing existing data.
- **Delete**: Removing data or records from the database.

Data Modelling

Data modelling involves creating abstract models that represent how data is structured within a database or a system. It is a process of defining the relationships between different data elements and setting rules to ensure data integrity. Data models are typically visualised through diagrams and are crucial in planning the architecture of databases, ensuring they are efficient, accurate, and scalable.

I started by downloading all six provided CSV files:

employees.csv:

On my first inspection, I found that the emp_no column was not in either ascending or descending order. I also found that the two date columns, birth_date and high_date values were of different data types. Some dates were recorded as MM/DD/YYYY and others as DD/MM/YYYY. One of the instructions required that we make a list of employees who had been hired in 1986.

At one first inspection, I couldn’t see whether any column would make a PRIMARY KEY as I wasn’t sure whether there would be any duplicates.

salaries.csv:

Unlike *employees.csv*, *emp_no* values were in ascending order. I wasn’t sure whether there were any duplicates.

I loaded employees.csv into Excel and did the following:

- Sorted *emp_num* into ascending order.

- I copied emp_no from salaries to employees and noted that they columns were the same length.
- Created a new column, *emp_id* as the PRIMARY KEY in employees, ascending from 1.
- Created a new column, *sal_id*, as a FOREIGN KEY in *employees.csv.*
- Created a new column called *hire_year,* to receive the year of hire later.
- I loaded *salaries.csv* into Excel and added a column as *sal_id* to act as the PRIMARY KEY in the salaries.csv table.

Changing Date Format and adding hire_year values in employees.csv

I changed all the date values *in birth_date* and *hire_date* columns to the DATE type. I extracted the *hire_year* from the *hire_date* and stored these values in the *hire_year* column.

titles.csv:

I loaded *titles.csv* into Excel and added a column *title_id* as the PRIMARY KEY for this table.

departments.csv:

I loaded ***deparments.csv*** into Excel and added a column *dept_id* as the PRIMARY KEY for this table.

dept_emp.csv:

I loaded ***dept_emp.csv*** into Excel and added a column *de_id* as the PRIMARY KEY for this table.

dept_manager.csv:

I didn’t change anything in *dept_manager.csv*

I was happy with the changes I had made.

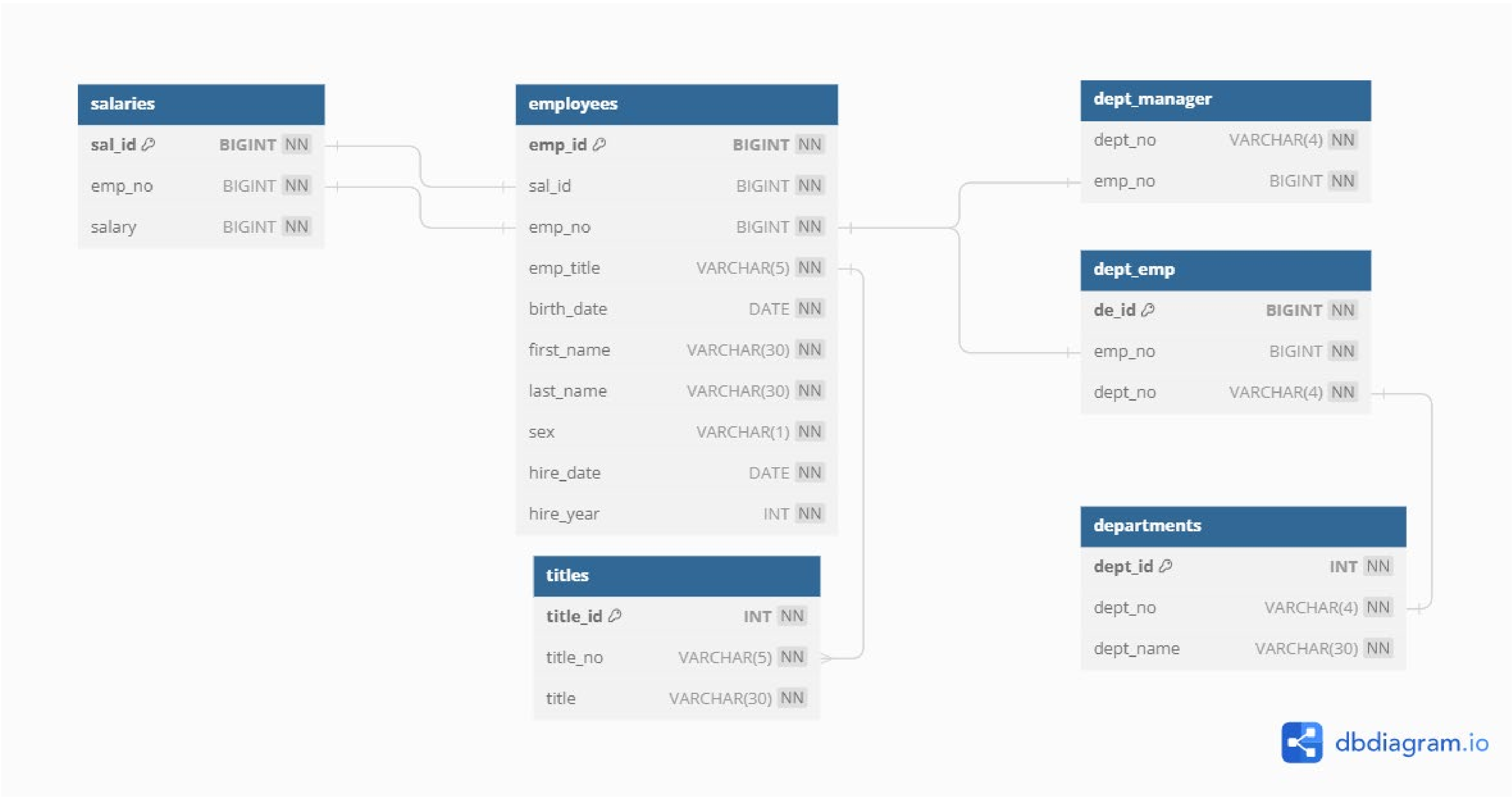
Constructing an "Entity-Relationship Diagram (ERD)."

The image is on the next page.

I constructed the image in dbdiagram.io. Click on the link to see the original work.

https://dbdiagram.io/d/Data-Modelling-ERD-65af5d71ac844320ae806b01

Entity Relationship Diagram - Challenge 9 SQL



salaries		
sal_id	**BIGINT**	NN
emp_no	BIGINT	NN
salary	BIGINT	NN

employees		
emp_id	**BIGINT**	NN
sal_id	BIGINT	NN
emp_no	BIGINT	NN
emp_title	VARCHAR(5)	NN
birth_date	DATE	NN
first_name	VARCHAR(30)	NN
last_name	VARCHAR(30)	NN
sex	VARCHAR(1)	NN
hire_date	DATE	NN
hire_year	INT	NN

titles		
title_id	**INT**	NN
title_no	VARCHAR(5)	NN
title	VARCHAR(30)	NN

dept_manager		
dept_no	VARCHAR(4)	NN
emp_no	BIGINT	NN

dept_emp		
de_id	**BIGINT**	NN
emp_no	BIGINT	NN
dept_no	VARCHAR(4)	NN

departments		
dept_id	**INT**	NN
dept_no	VARCHAR(4)	NN
dept_name	VARCHAR(30)	NN

Data Engineering

Data engineering focuses on the practical application of data collection, storage, and retrieval. It involves building and maintaining the infrastructure and architecture, allowing large-scale data processing and analysis. This includes setting up databases, data warehousing, ensuring efficient data pipelines, and handling big data technologies. Data engineers ensure data is accessible, reliable, and formatted for analysis by data scientists and analysts.

Below I have addressed all of the tasks in the Data Engineering section of Challenge 9

1 All required columns are defined for each table

a **employees**

- i emp_id bigint PRIMARY KEY not null
- ii sal_id bigint FOREIGN KEY not null
- iii emp_no bigint not null
- iv emp_title varchar(5) not null
- v Birth_date DATE not null
- vi first_name varchar(30) not null
- vii last_name varchar(30) not null
- viii sex varchar(1) not null
- ix hire_date DATE not_null
- x Hire_year int not null

b **Salaries**

- I Sal_id bigint PRIMARY KEY not null
- Ii Emp_no bigint not null
- Iii Salary bigint not null

c **titles**

- I Title_id int PRIMARY KEY not null
- Ii Title_no varchar(5) not null
- iii Title varchar(30) not null

d departments

- a dept_id int PRIMARY KEY not null
- b dept_no varchar(4) not null
- c dept_name varchar(30) not null

e **Dept_emp**

- i de_id bigint PRIMARY KEY

ii emp_no bigint not null

iii dept_no varchar(4) not null

f Dept_manager

i dept_no varchar(4) not null

ii emp_no bigint not null

2 Columns are set to the correct data type

The correct data type for columns is important in data analysis, including different data types of operations, efficient memory usage, and appropriate functionality and methods.

I have included all the column data types in my table above. I believe that they are all correct.

3 Primary Keys are set for each table

Setting primary keys for each table in a database is important for several reasons, including ensuring every record in the table is unique, indexing, establishing a relationship between tables, data integrity, and efficient updates and deletions. They are fundamental in relational databases.

I have set PRIMARY KEYS for 5 tables. I did not set one column as the PRIMARY KEY in dept_manager. A combination of dept_no and emp_no will always be unique.

4 Correctly references related tables

Correctly referencing related tables in a database ensures that relationships between tables are consistently maintained. They also allow for more straightforward and efficient queries and simplify complex relations

From my Entity Relationship Diagram, you will see that I have tried to correctly reference related tables by focusing on how the tables are interconnected through relationship keys. I started by recognising the relationship between tables as defined by my ERD diagram. I noticed that some JOIN queries would be required in the data analysis section.

5 Tables are correctly related using Foreign Keys.

The reasons for relating tables correctly using FOREIGN KEYS include enforcing referential integrity by ensuring that a value in the child table corresponds to a valid existing record in the parent table, ensuring data consistency, facilitating the Join of tables, and simplifying data retrieval and analysis.

From my Entity Relationship Diagram, you will see that I have tried to correctly reference related tables by focusing on how the tables are interconnected through relationship keys. I started by recognising the relationship between tables as defined by my ERD diagram.

6 Correctly users the NOT NULL condition necessary columns

NOT NULL ensures that a column can’t have a null value. It prevents inaccurate data or assumptions. It facilitates data analysis.

However, it can reduce flexibility by forcing values into columns where data may not be meaningful.

I decided that column should be NOT NULL as my inspection of the table data in the csv files indicated that all data column values should be included to facilitate the data analysis.

7 Accurately defines value length for columns.

Accurately defining the value length for columns in a database is important for several reasons, including data integrity, efficient storage, optimising storage space, performance optimisation, validation and quality control, avoiding data truncation, and for legal and compliance reasons.

Refer to my ERD to see that I defined the value length for all columns.

Q

Data Analysis

Data analysis involves examining, cleaning, transforming, and modelling data to discover useful information, inform conclusions, and support decision-making. It involves using statistical, algorithmic, mining, or visualisation techniques to interpret and present data. Data analysis is key in identifying trends, patterns, and insights that can lead to meaningful and actionable business, scientific, or social conclusions.

I created one SQL file, which I have included in this submission:

Challenge9-data_analysis.sql

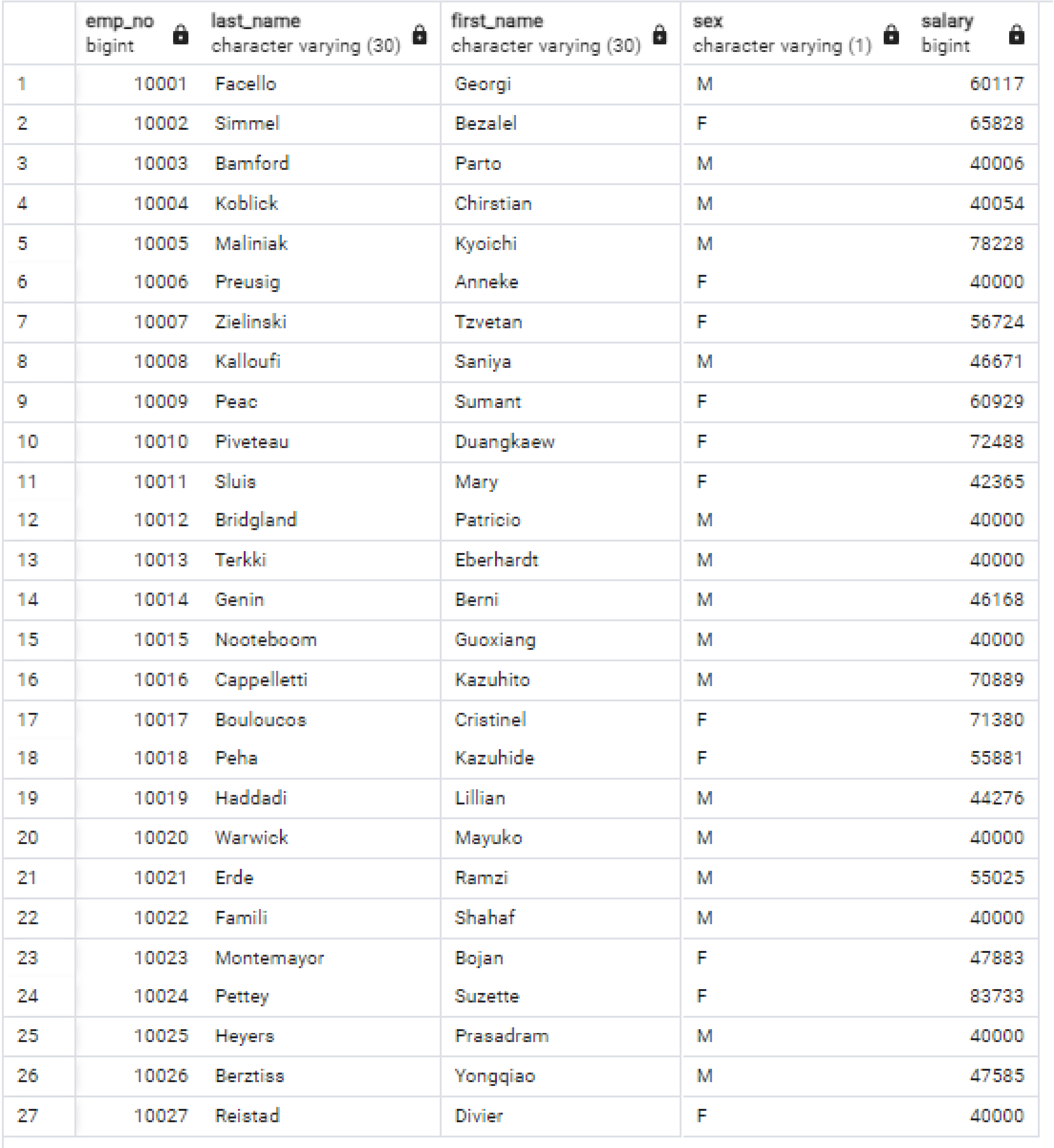
I have addressed each Data Analysis item below, including the SQL code and output.

1 List the employee number, last name, first name, sex and salary of each employee.

SQL Code

```
SELECT e.emp_no, e.last_name, e.first_name, e.sex, s.salary
FROM employees e
JOIN salaries s ON e.emp_no = s.emp_no
LIMIT 25;
```

Output



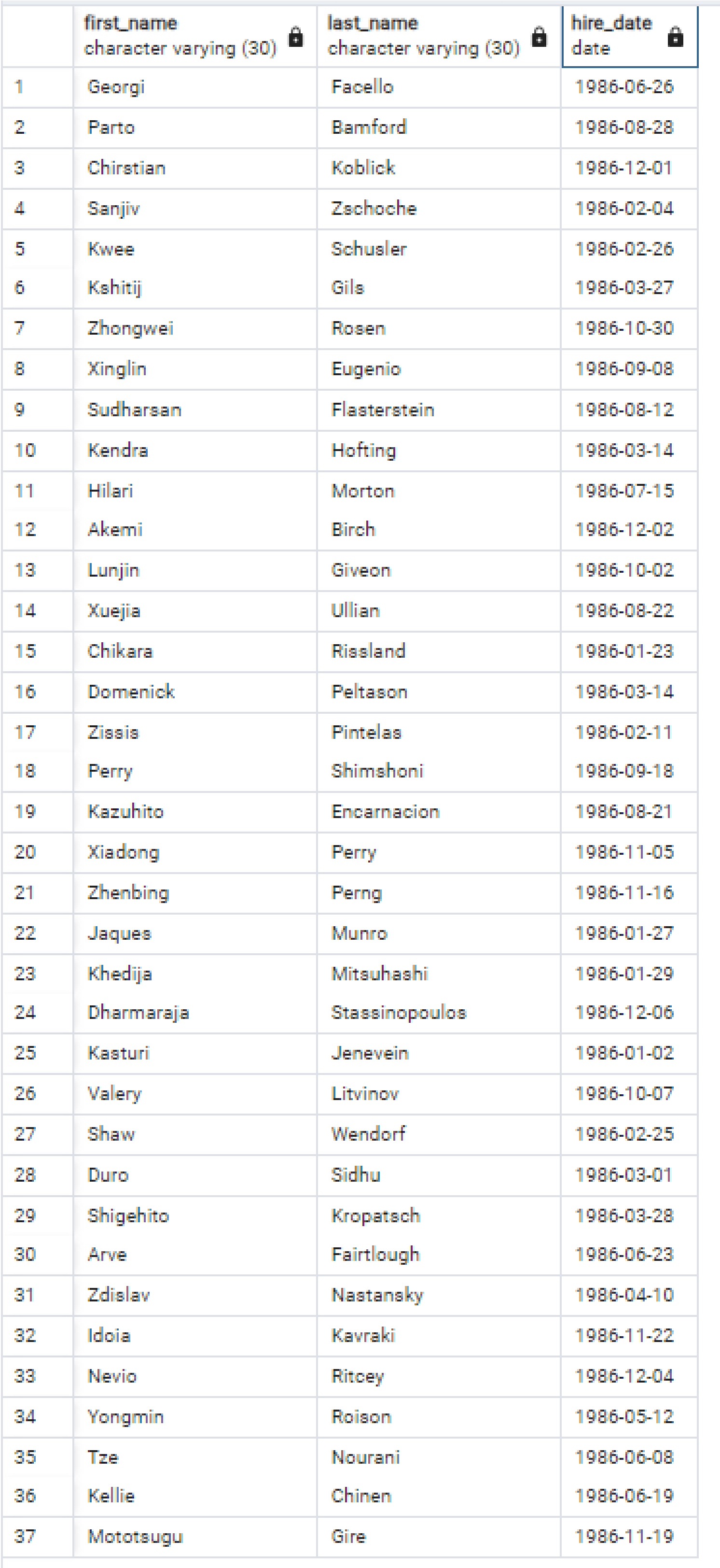
	emp_no bigint	last_name character varying (30)	first_name character varying (30)	sex character varying (1)	salary bigint
1	10001	Facello	Georgi	M	60117
2	10002	Simmel	Bezalel	F	65828
3	10003	Bamford	Parto	M	40006
4	10004	Koblick	Chirstian	M	40054
5	10005	Maliniak	Kyoichi	M	78228
6	10006	Preusig	Anneke	F	40000
7	10007	Zielinski	Tzvetan	F	56724
8	10008	Kalloufi	Saniya	M	46671
9	10009	Peac	Sumant	F	60929
10	10010	Piveteau	Duangkaew	F	72488
11	10011	Sluis	Mary	F	42365
12	10012	Bridgland	Patricio	M	40000
13	10013	Terkki	Eberhardt	M	40000
14	10014	Genin	Berni	M	46168
15	10015	Nooteboom	Guoxiang	M	40000
16	10016	Cappelletti	Kazuhito	M	70889
17	10017	Bouloucos	Cristinel	F	71380
18	10018	Peha	Kazuhide	F	55881
19	10019	Haddadi	Lillian	M	44276
20	10020	Warwick	Mayuko	M	40000
21	10021	Erde	Ramzi	M	55025
22	10022	Famili	Shahaf	M	40000
23	10023	Montemayor	Bojan	F	47883
24	10024	Pettey	Suzette	F	83733
25	10025	Heyers	Prasadram	M	40000
26	10026	Berztiss	Yongqiao	M	47585
27	10027	Reistad	Divier	F	40000

2 List the first name, last name, and hire date for for the employees who were hired in 1986.

SQL Code

```
SELECT first_name, last_name, hire_date
FROM employees
WHERE hire_year = 1986
LIMIT 37;
```

Output



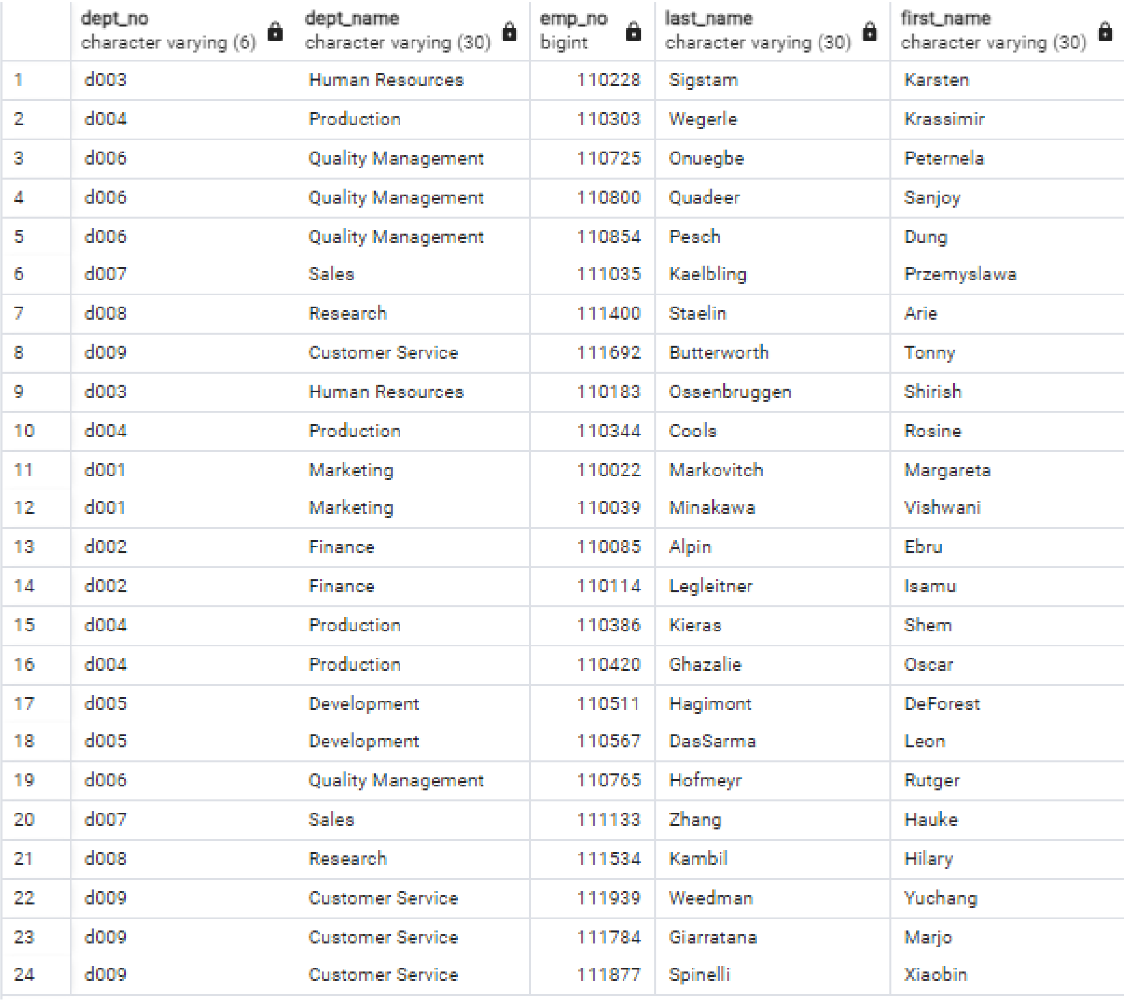
	first_name character varying (30)	last_name character varying (30)	hire_date date
1	Georgi	Facello	1986-06-26
2	Parto	Bamford	1986-08-28
3	Chirstian	Koblick	1986-12-01
4	Sanjiv	Zschoche	1986-02-04
5	Kwee	Schusler	1986-02-26
6	Kshitij	Gils	1986-03-27
7	Zhongwei	Rosen	1986-10-30
8	Xinglin	Eugenio	1986-09-08
9	Sudharsan	Flasterstein	1986-08-12
10	Kendra	Hofting	1986-03-14
11	Hilari	Morton	1986-07-15
12	Akemi	Birch	1986-12-02
13	Lunjin	Giveon	1986-10-02
14	Xuejia	Ullian	1986-08-22
15	Chikara	Rissland	1986-01-23
16	Domenick	Peltason	1986-03-14
17	Zissis	Pintelas	1986-02-11
18	Perry	Shimshoni	1986-09-18
19	Kazuhito	Encarnacion	1986-08-21
20	Xiadong	Perry	1986-11-05
21	Zhenbing	Perng	1986-11-16
22	Jaques	Munro	1986-01-27
23	Khedija	Mitsuhashi	1986-01-29
24	Dharmaraja	Stassinopoulos	1986-12-06
25	Kasturi	Jenevein	1986-01-02
26	Valery	Litvinov	1986-10-07
27	Shaw	Wendorf	1986-02-25
28	Duro	Sidhu	1986-03-01
29	Shigehito	Kropatsch	1986-03-28
30	Arve	Fairtlough	1986-06-23
31	Zdislav	Nastansky	1986-04-10
32	Idoia	Kavraki	1986-11-22
33	Nevio	Ritcey	1986-12-04
34	Yongmin	Roison	1986-05-12
35	Tze	Nourani	1986-06-08
36	Kellie	Chinen	1986-06-19
37	Mototsugu	Gire	1986-11-19

3 List the manager of each department along with their department number

SQL Code

```
SELECT dm.dept_no, d.dept_name, dm.emp_no, e.last_name, e.first_name
FROM dept_manager dm
JOIN departments d ON dm.dept_no = d.dept_no
JOIN employees e ON dm.emp_no = e.emp_no;
```

Output



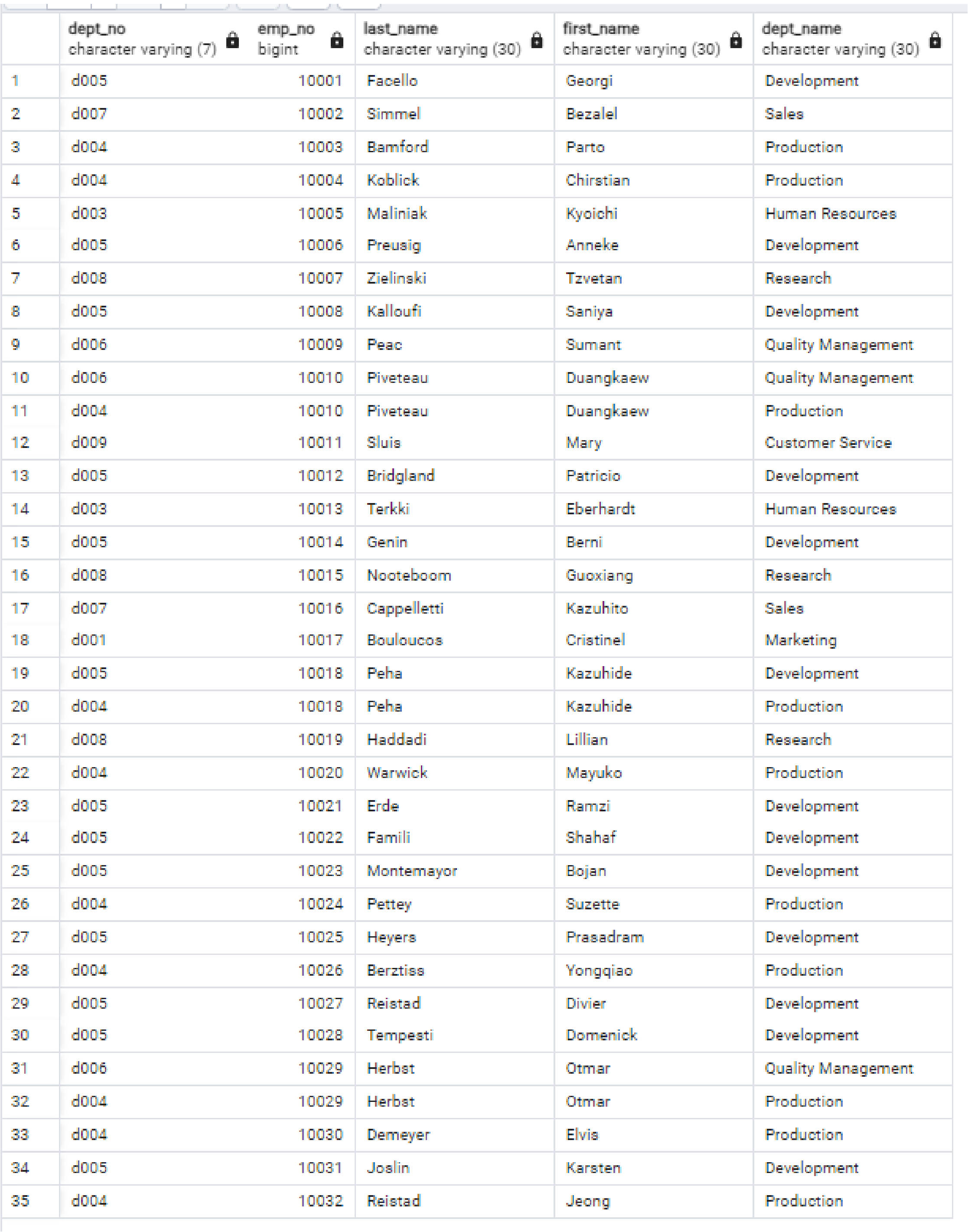
	dept_no character varying (6)	dept_name character varying (30)	emp_no bigint	last_name character varying (30)	first_name character varying (30)
1	d003	Human Resources	110228	Sigstam	Karsten
2	d004	Production	110303	Wegerle	Krassimir
3	d006	Quality Management	110725	Onuegbe	Peternela
4	d006	Quality Management	110800	Quadeer	Sanjoy
5	d006	Quality Management	110854	Pesch	Dung
6	d007	Sales	111035	Kaelbling	Przemyslawa
7	d008	Research	111400	Staelin	Arie
8	d009	Customer Service	111692	Butterworth	Tonny
9	d003	Human Resources	110183	Ossenbruggen	Shirish
10	d004	Production	110344	Cools	Rosine
11	d001	Marketing	110022	Markovitch	Margareta
12	d001	Marketing	110039	Minakawa	Vishwani
13	d002	Finance	110085	Alpin	Ebru
14	d002	Finance	110114	Legleitner	Isamu
15	d004	Production	110386	Kieras	Shem
16	d004	Production	110420	Ghazalie	Oscar
17	d005	Development	110511	Hagimont	DeForest
18	d005	Development	110567	DasSarma	Leon
19	d006	Quality Management	110765	Hofmeyr	Rutger
20	d007	Sales	111133	Zhang	Hauke
21	d008	Research	111534	Kambil	Hilary
22	d009	Customer Service	111939	Weedman	Yuchang
23	d009	Customer Service	111784	Giarratana	Marjo
24	d009	Customer Service	111877	Spinelli	Xiaobin

4 **List the department number for each employee along with that employee's employee number, last name, first name, and department name.**

SQL Code

```
SELECT de.dept_no, e.emp_no, e.last_name, e.first_name, d.dept_name
FROM employees e
JOIN dept_emp de ON e.emp_no = de.emp_no
JOIN departments d ON de.dept_no = d.dept_no
LIMIT 20;
```

Output



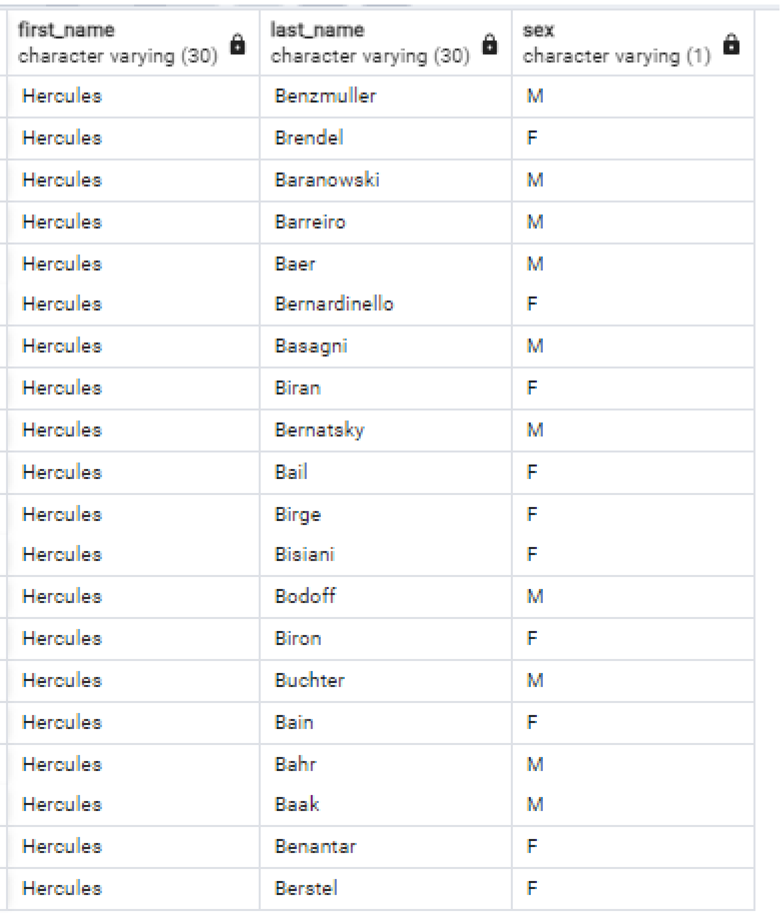
	dept_no character varying (7)	emp_no bigint	last_name character varying (30)	first_name character varying (30)	dept_name character varying (30)
1	d005	10001	Facello	Georgi	Development
2	d007	10002	Simmel	Bezalel	Sales
3	d004	10003	Bamford	Parto	Production
4	d004	10004	Koblick	Chirstian	Production
5	d003	10005	Maliniak	Kyoichi	Human Resources
6	d005	10006	Preusig	Anneke	Development
7	d008	10007	Zielinski	Tzvetan	Research
8	d005	10008	Kalloufi	Saniya	Development
9	d006	10009	Peac	Sumant	Quality Management
10	d006	10010	Piveteau	Duangkaew	Quality Management
11	d004	10010	Piveteau	Duangkaew	Production
12	d009	10011	Sluis	Mary	Customer Service
13	d005	10012	Bridgland	Patricio	Development
14	d003	10013	Terkki	Eberhardt	Human Resources
15	d005	10014	Genin	Berni	Development
16	d008	10015	Nooteboom	Guoxiang	Research
17	d007	10016	Cappelletti	Kazuhito	Sales
18	d001	10017	Bouloucos	Cristinel	Marketing
19	d005	10018	Peha	Kazuhide	Development
20	d004	10018	Peha	Kazuhide	Production
21	d008	10019	Haddadi	Lillian	Research
22	d004	10020	Warwick	Mayuko	Production
23	d005	10021	Erde	Ramzi	Development
24	d005	10022	Famili	Shahaf	Development
25	d005	10023	Montemayor	Bojan	Development
26	d004	10024	Pettey	Suzette	Production
27	d005	10025	Heyers	Prasadram	Development
28	d004	10026	Berztiss	Yongqiao	Production
29	d005	10027	Reistad	Divier	Development
30	d005	10028	Tempesti	Domenick	Development
31	d006	10029	Herbst	Otmar	Quality Management
32	d004	10029	Herbst	Otmar	Production
33	d004	10030	Demeyer	Elvis	Production
34	d005	10031	Joslin	Karsten	Development
35	d004	10032	Reistad	Jeong	Production

5 List the first name, last name, and sex of each employee whose first name is Hercules and whose last name begins with the letter B.

SQL Code

```
SELECT first_name, last_name, sex
FROM employees
WHERE first_name = 'Hercules' AND last_name LIKE 'B%';
```

Output



first_name character varying (30)	last_name character varying (30)	sex character varying (1)
Hercules	Benzmuller	M
Hercules	Brendel	F
Hercules	Baranowski	M
Hercules	Barreiro	M
Hercules	Baer	M
Hercules	Bernardinello	F
Hercules	Basagni	M
Hercules	Biran	F
Hercules	Bernatsky	M
Hercules	Bail	F
Hercules	Birge	F
Hercules	Bisiani	F
Hercules	Bodoff	M
Hercules	Biron	F
Hercules	Buchter	M
Hercules	Bain	F
Hercules	Bahr	M
Hercules	Baak	M
Hercules	Benantar	F
Hercules	Berstel	F

6 List each employee in the Sales Department, including their employee number, last name, and first name.

SQL Code

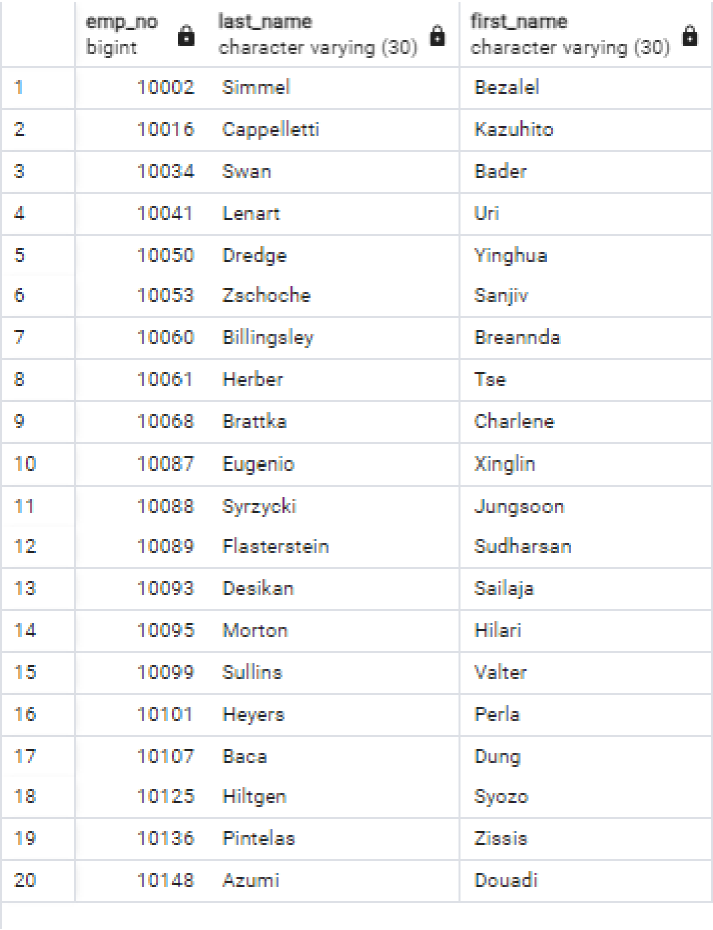
List names

```
SELECT e.emp_no, e.last_name, e.first_name
FROM employees e
JOIN dept_emp de ON e.emp_no = de.emp_no
JOIN departments d ON de.dept_no = d.dept_no
WHERE d.dept_name = 'Sales'
LIMIT 20;
```

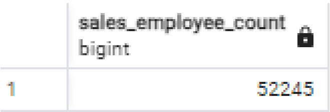
Count the number of people in the Sales Department (I am interested to know

```
SELECT COUNT(DISTINCT e.emp_no) AS Sales_Employee_Count
FROM employees e
JOIN dept_emp de ON e.emp_no = de.emp_no
JOIN departments d ON de.dept_no = d.dept_no
WHERE d.dept_name = 'Sales';
```

Output



	emp_no bigint	last_name character varying (30)	first_name character varying (30)
1	10002	Simmel	Bezalel
2	10016	Cappelletti	Kazuhito
3	10034	Swan	Bader
4	10041	Lenart	Uri
5	10050	Dredge	Yinghua
6	10053	Zschoche	Sanjiv
7	10060	Billingsley	Breannda
8	10061	Herber	Tse
9	10068	Brattka	Charlene
10	10087	Eugenio	Xinglin
11	10088	Syrzycki	Jungsoon
12	10089	Flasterstein	Sudharsan
13	10093	Desikan	Sailaja
14	10095	Morton	Hilari
15	10099	Sullins	Valter
16	10101	Heyers	Perla
17	10107	Baca	Dung
18	10125	Hiltgen	Syozo
19	10136	Pintelas	Zissis
20	10148	Azumi	Douadi



	sales_employee_count bigint
1	52245

I counted 52,245 people in the Sales Department

7 **List each employee in the Sales and Development departments, including their employee number, last name, first name, and department name.**

SQL Code

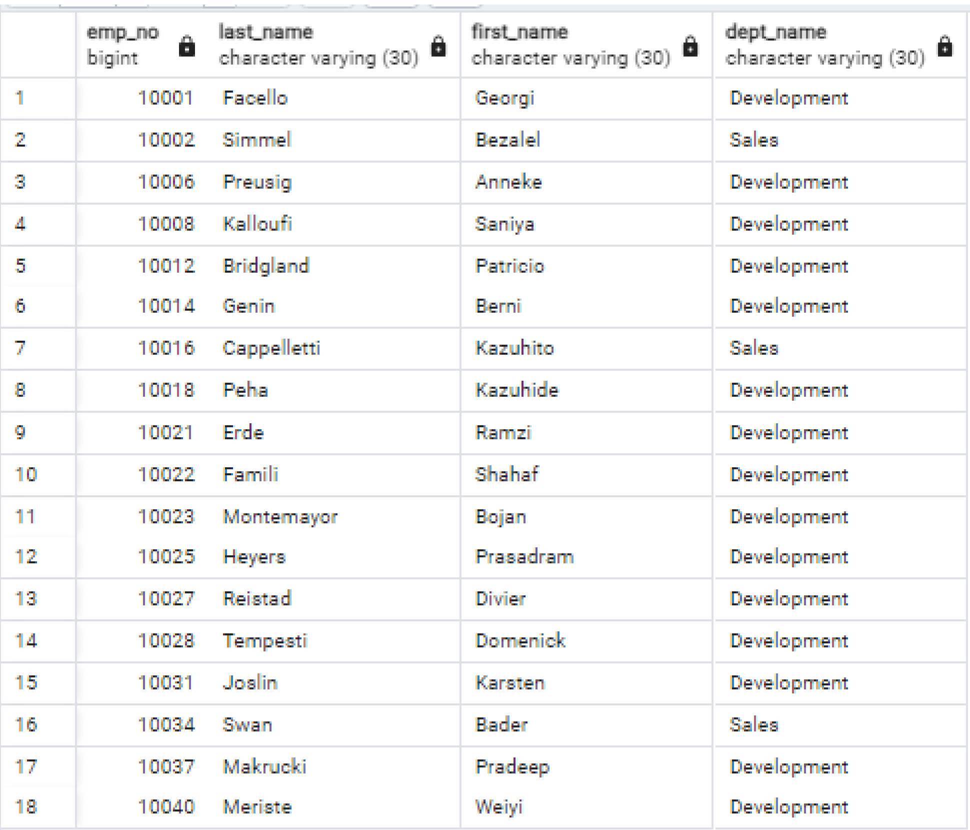
Find the people the people in the Sales and Development Departments

```
SELECT e.emp_no, e.last_name, e.first_name, d.dept_name
FROM employees e
JOIN dept_emp de ON e.emp_no = de.emp_no
JOIN departments d ON de.dept_no = d.dept_no
WHERE d.dept_name IN ('Sales', 'Development');
```

Counting the number of people in each department (For my interest)

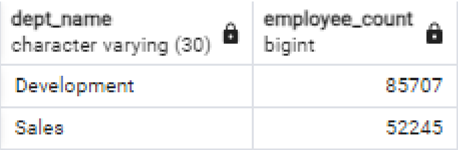
```
SELECT d.dept_name, COUNT(DISTINCT e.emp_no) AS Employee_Count
FROM employees e
JOIN dept_emp de ON e.emp_no = de.emp_no
JOIN departments d ON de.dept_no = d.dept_no
WHERE d.dept_name IN ('Sales', 'Development')
GROUP BY d.dept_name;
```

Output



	emp_no bigint	last_name character varying (30)	first_name character varying (30)	dept_name character varying (30)
1	10001	Facello	Georgi	Development
2	10002	Simmel	Bezalel	Sales
3	10006	Preusig	Anneke	Development
4	10008	Kalloufi	Saniya	Development
5	10012	Bridgland	Patricio	Development
6	10014	Genin	Berni	Development
7	10016	Cappelletti	Kazuhito	Sales
8	10018	Peha	Kazuhide	Development
9	10021	Erde	Ramzi	Development
10	10022	Famili	Shahaf	Development
11	10023	Montemayor	Bojan	Development
12	10025	Heyers	Prasadram	Development
13	10027	Reistad	Divier	Development
14	10028	Tempesti	Domenick	Development
15	10031	Joslin	Karsten	Development
16	10034	Swan	Bader	Sales
17	10037	Makrucki	Pradeep	Development
18	10040	Meriste	Weiyi	Development

Output 7A



dept_name character varying (30)	employee_count bigint
Development	85707
Sales	52245

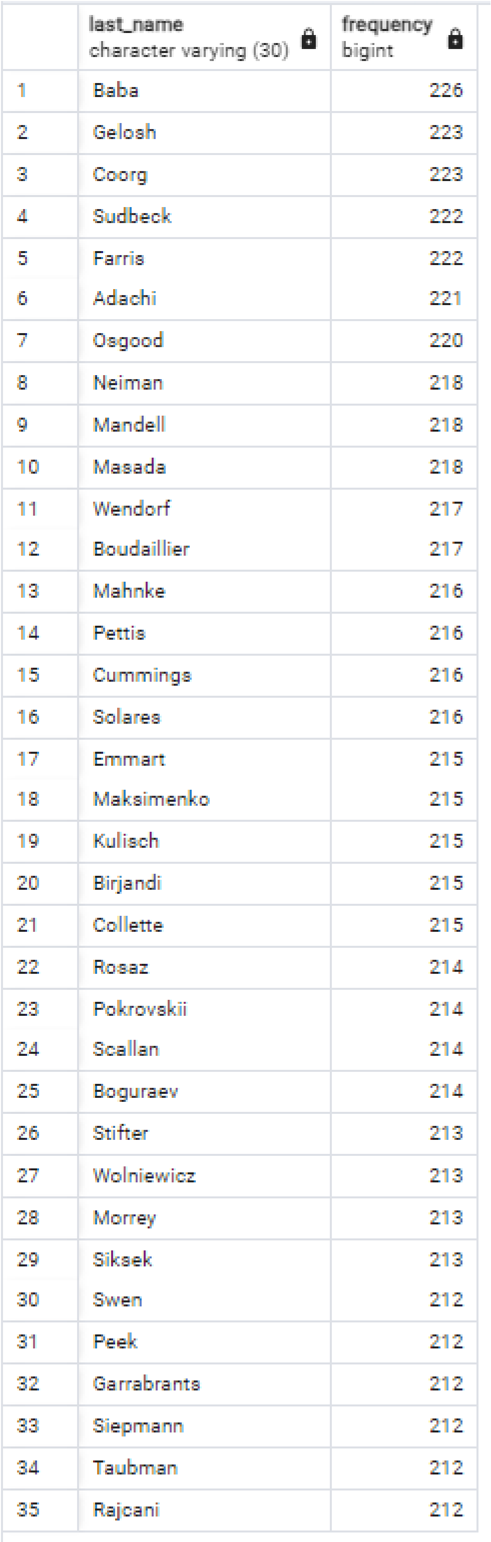
Count the number of people in each of the Sales and Development departments.

8 List the frequency counts, in descending order, of all the employee last names (that is, how many employees share each last name)

SQL Code

```
SELECT last_name, COUNT(emp_no) AS Frequency
FROM employees
GROUP BY last_name
ORDER BY Frequency DESC
LIMIT 35;
```

Output



	last_name character varying (30)	frequency bigint
1	Baba	226
2	Gelosh	223
3	Coorg	223
4	Sudbeck	222
5	Farris	222
6	Adachi	221
7	Osgood	220
8	Neiman	218
9	Mandell	218
10	Masada	218
11	Wendorf	217
12	Boudaillier	217
13	Mahnke	216
14	Pettis	216
15	Cummings	216
16	Solares	216
17	Emmart	215
18	Maksimenko	215
19	Kulisch	215
20	Birjandi	215
21	Collette	215
22	Rosaz	214
23	Pokrovskii	214
24	Scallan	214
25	Boguraev	214
26	Stifter	213
27	Wolniewicz	213
28	Morrey	213
29	Siksek	213
30	Swen	212
31	Peek	212
32	Garrabrants	212
33	Siepmann	212
34	Taubman	212
35	Rajcani	212