**The challenges**

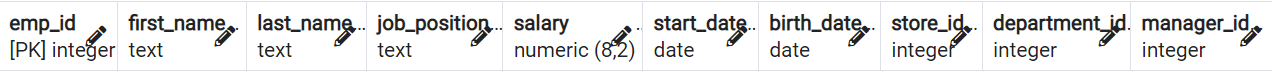
**Task 1**

Difficulty: Moderate

**Task 1.1**

In your company there hasn't been a database table with all the employee information yet.

You need to set up the table called employees in the following way:



There should be NOT NULL constraints for the following columns:

first\_name,

last\_name ,

job\_position,

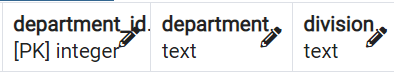
start\_date DATE,

birth\_date DATE

1. CREATE TABLE employees (
2. emp\_id SERIAL PRIMARY KEY,
3. first\_name TEXT NOT NULL,
4. last\_name TEXT NOT NULL,
5. job\_position TEXT NOT NULL,
6. salary decimal(8,2),
7. start\_date DATE NOT NULL,
8. birth\_date DATE NOT NULL,
9. store\_id INT REFERENCES store(store\_id),
10. department\_id INT,
11. manager\_id INT
12. );

**Task 1.2**

Set up an additional table called departments in the following way:



Additionally no column should allow nulls.

1. CREATE TABLE departments (
2. department\_id SERIAL PRIMARY KEY,
3. department TEXT NOT NULL,
4. division TEXT NOT NULL);

**Task 2**

Difficulty: Moderate

Alter the employees table in the following way:

- Set the column department\_id to not null.

- Add a default value of CURRENT\_DATE to the column start\_date.

- Add the column end\_date with an appropriate data type (one that you think makes sense).

- Add a constraint called birth\_check that doesn't allow birth dates that are in the future.

- Rename the column job\_position to position\_title.

1. ALTER TABLE employees
2. ALTER COLUMN department\_id SET NOT NULL,
3. ALTER COLUMN start\_date SET DEFAULT CURRENT\_DATE,
4. ADD COLUMN end\_date DATE,
5. ADD CONSTRAINT birth\_check CHECK(birth\_date < CURRENT\_DATE);
6. ALTER TABLE employees
7. RENAME job\_position TO position\_title;

**Task 3**

Difficulty: Moderate

1. INSERT INTO employees
2. VALUES
3. (1,'Morrie','Conaboy','CTO',21268.94,'2005-04-30','1983-07-10',1,1,NULL,NULL),
4. (2,'Miller','McQuarter','Head of BI',14614.00,'2019-07-23','1978-11-09',1,1,1,NULL),
5. (3,'Christalle','McKenny','Head of Sales',12587.00,'1999-02-05','1973-01-09',2,3,1,NULL),
6. (4,'Sumner','Seares','SQL Analyst',9515.00,'2006-05-31','1976-08-03',2,1,6,NULL),
7. (5,'Romain','Hacard','BI Consultant',7107.00,'2012-09-24','1984-07-14',1,1,6,NULL),
8. (6,'Ely','Luscombe','Team Lead Analytics',12564.00,'2002-06-12','1974-08-01',1,1,2,NULL),
9. (7,'Clywd','Filyashin','Senior SQL Analyst',10510.00,'2010-04-05','1989-07-23',2,1,2,NULL),
10. (8,'Christopher','Blague','SQL Analyst',9428.00,'2007-09-30','1990-12-07',2,2,6,NULL),
11. (9,'Roddie','Izen','Software Engineer',4937.00,'2019-03-22','2008-08-30',1,4,6,NULL),
12. (10,'Ammamaria','Izhak','Customer Support',2355.00,'2005-03-17','1974-07-27',2,5,3,'2013-04-14'),
13. (11,'Carlyn','Stripp','Customer Support',3060.00,'2013-09-06','1981-09-05',1,5,3,NULL),
14. (12,'Reuben','McRorie','Software Engineer',7119.00,'1995-12-31','1958-08-15',1,5,6,NULL),
15. (13,'Gates','Raison','Marketing Specialist',3910.00,'2013-07-18','1986-06-24',1,3,3,NULL),
16. (14,'Jordanna','Raitt','Marketing Specialist',5844.00,'2011-10-23','1993-03-16',2,3,3,NULL),
17. (15,'Guendolen','Motton','BI Consultant',8330.00,'2011-01-10','1980-10-22',2,3,6,NULL),
18. (16,'Doria','Turbat','Senior SQL Analyst',9278.00,'2010-08-15','1983-01-11',1,1,6,NULL),
19. (17,'Cort','Bewlie','Project Manager',5463.00,'2013-05-26','1986-10-05',1,5,3,NULL),
20. (18,'Margarita','Eaden','SQL Analyst',5977.00,'2014-09-24','1978-10-08',2,1,6,'2020-03-16'),
21. (19,'Hetty','Kingaby','SQL Analyst',7541.00,'2009-08-17','1999-04-25',1,2,6,NULL),
22. (20,'Lief','Robardley','SQL Analyst',8981.00,'2002-10-23','1971-01-25',2,3,6,'2016-07-01'),
23. (21,'Zaneta','Carlozzi','Working Student',1525.00,'2006-08-29','1995-04-16',1,3,6,'2012-02-19'),
24. (22,'Giana','Matz','Working Student',1036.00,'2016-03-18','1987-09-25',1,3,6,NULL),
25. (23,'Hamil','Evershed','Web Developper',3088.00,'2022-02-03','2012-03-30',1,4,2,NULL),
26. (24,'Lowe','Diamant','Web Developper',6418.00,'2018-12-31','2002-09-07',1,4,2,NULL),
27. (25,'Jack','Franklin','SQL Analyst',6771.00,'2013-05-18','2005-10-04',1,2,2,NULL),
28. (26,'Jessica','Brown','SQL Analyst',8566.00,'2003-10-23','1965-01-29',1,1,2,NULL);

**Task 3.1**

Insert the following values into the *employees*table.

There will be most likely an error when you try to insert the values.

So, try to insert the values and then fix the error.

*Columns:*

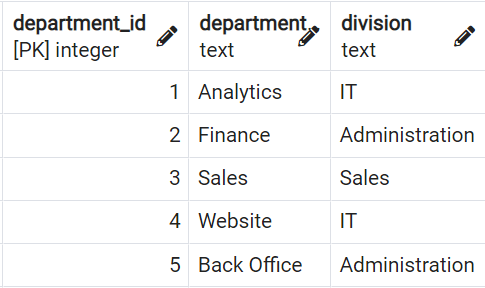
1. (emp\_id,first\_name,last\_name,position\_title,salary,start\_date,birth\_date,store\_id,department\_id,manager\_id,end\_date)

*Values:*

1. (1,'Morrie','Conaboy','CTO',21268.94,'2005-04-30','1983-07-10',1,1,NULL,NULL),
2. (2,'Miller','McQuarter','Head of BI',14614.00,'2019-07-23','1978-11-09',1,1,1,NULL),
3. (3,'Christalle','McKenny','Head of Sales',12587.00,'1999-02-05','1973-01-09',2,3,1,NULL),
4. (4,'Sumner','Seares','SQL Analyst',9515.00,'2006-05-31','1976-08-03',2,1,6,NULL),
5. (5,'Romain','Hacard','BI Consultant',7107.00,'2012-09-24','1984-07-14',1,1,6,NULL),
6. (6,'Ely','Luscombe','Team Lead Analytics',12564.00,'2002-06-12','1974-08-01',1,1,2,NULL),
7. (7,'Clywd','Filyashin','Senior SQL Analyst',10510.00,'2010-04-05','1989-07-23',2,1,2,NULL),
8. (8,'Christopher','Blague','SQL Analyst',9428.00,'2007-09-30','1990-12-07',2,2,6,NULL),
9. (9,'Roddie','Izen','Software Engineer',4937.00,'2019-03-22','2008-08-30',1,4,6,NULL),
10. (10,'Ammamaria','Izhak','Customer Support',2355.00,'2005-03-17','1974-07-27',2,5,3,2013-04-14),
11. (11,'Carlyn','Stripp','Customer Support',3060.00,'2013-09-06','1981-09-05',1,5,3,NULL),
12. (12,'Reuben','McRorie','Software Engineer',7119.00,'1995-12-31','1958-08-15',1,5,6,NULL),
13. (13,'Gates','Raison','Marketing Specialist',3910.00,'2013-07-18','1986-06-24',1,3,3,NULL),
14. (14,'Jordanna','Raitt','Marketing Specialist',5844.00,'2011-10-23','1993-03-16',2,3,3,NULL),
15. (15,'Guendolen','Motton','BI Consultant',8330.00,'2011-01-10','1980-10-22',2,3,6,NULL),
16. (16,'Doria','Turbat','Senior SQL Analyst',9278.00,'2010-08-15','1983-01-11',1,1,6,NULL),
17. (17,'Cort','Bewlie','Project Manager',5463.00,'2013-05-26','1986-10-05',1,5,3,NULL),
18. (18,'Margarita','Eaden','SQL Analyst',5977.00,'2014-09-24','1978-10-08',2,1,6,2020-03-16),
19. (19,'Hetty','Kingaby','SQL Analyst',7541.00,'2009-08-17','1999-04-25',1,2,6,'NULL'),
20. (20,'Lief','Robardley','SQL Analyst',8981.00,'2002-10-23','1971-01-25',2,3,6,2016-07-01),
21. (21,'Zaneta','Carlozzi','Working Student',1525.00,'2006-08-29','1995-04-16',1,3,6,2012-02-19),
22. (22,'Giana','Matz','Working Student',1036.00,'2016-03-18','1987-09-25',1,3,6,NULL),
23. (23,'Hamil','Evershed','Web Developper',3088.00,'2022-02-03','2012-03-30',1,4,2,NULL),
24. (24,'Lowe','Diamant','Web Developper',6418.00,'2018-12-31','2002-09-07',1,4,2,NULL),
25. (25,'Jack','Franklin','SQL Analyst',6771.00,'2013-05-18','2005-10-04',1,2,2,NULL),
26. (26,'Jessica','Brown','SQL Analyst',8566.00,'2003-10-23','1965-01-29',1,1,2,NULL)

**Task 3.2**

Insert the following values into the departments table.



1. INSERT INTO departments
2. VALUES (1, 'Analytics','IT'),
3. (2, 'Finance','Administration'),
4. (3, 'Sales','Sales'),
5. (4, 'Website','IT'),
6. (5, 'Back Office','Administration')

**Task 4**

Difficulty: Moderate

**Task 4.1**

Jack Franklin gets promoted to 'Senior SQL Analyst' and the salary raises to 7200.

Update the values accordingly.

1. UPDATE employees
2. SET position\_title = 'Senior SQL Analyst'
3. WHERE emp\_id=25;
4. UPDATE employees
5. SET salary=7200
6. WHERE emp\_id=25;

**Task 4.2**

The responsible people decided to rename the position\_title Customer Support to Customer Specialist.

Update the values accordingly.

1. UPDATE employees
2. SET position\_title='Customer Specialist'
3. WHERE position\_title='Customer Support';

**Task 4.3**

All SQL Analysts including Senior SQL Analysts get a raise of 6%.

Upate the salaries accordingly.

1. UPDATE employees
2. SET salary=salary\*1.06
3. WHERE position\_title LIKE '%SQL Analyst';

**Task 4.4**

*Question:*

What is the average salary of a SQL Analyst in the company (excluding Senior SQL Analyst)?

*Answer:*

8834.75

1. SELECT ROUND(AVG(salary),2) FROM employees
2. WHERE position\_title='SQL Analyst'

**Task 5**

Difficulty: Advanced

**Task 5.1**

Write a query that adds a column called manager that contains  first\_name and last\_name (in one column) in the data output.

Secondly, add a column called is\_active with 'false' if the employee has left the company already, otherwise the value is 'true'.

1. SELECT
2. emp.\*,
3. CASE WHEN emp.end\_date IS NULL THEN 'true'
4. ELSE 'false'
5. END as is\_active,
6. mng.first\_name ||' '|| mng.last\_name AS manager\_name
7. FROM employees emp
8. LEFT JOIN employees mng
9. ON emp.manager\_id=mng.emp\_id;

**Task 5.2**

Create a view called v\_employees\_info from that previous query.

1. CREATE VIEW v\_employees\_info
2. AS
3. SELECT
4. emp.\*,
5. CASE WHEN emp.end\_date IS NULL THEN 'true'
6. ELSE 'false'
7. END as is\_active,
8. mng.first\_name ||' '|| mng.last\_name AS manager\_name
9. FROM employees emp
10. LEFT JOIN employees mng
11. ON emp.manager\_id=mng.emp\_id;

**Task 6**

Difficulty: Moderate

Write a query that returns the average salaries for each positions with appropriate roundings.

*Question:*

What is the average salary for a Software Engineer in the company.

*Answer:*

6028.00

1. SELECT
2. position\_title,
3. ROUND(AVG(salary),2)
4. FROM v\_employees\_info
5. GROUP BY position\_title
6. ORDER BY 2;

**Task 7**

Difficulty: Moderate

Write a query that returns the average salaries per division.

*Question:*

What is the average salary in the Sales department?

Answer:

6107.41

1. SELECT
2. division,
3. ROUND(AVG(salary),2)
4. FROM employees e
5. LEFT JOIN departments d
6. ON e.department\_id=d.department\_id
7. GROUP BY division
8. ORDER BY 2

**Task 8**

Difficulty: Advanced

**Task 8.1**

Write a query that returns the following:

emp\_id,

first\_name,

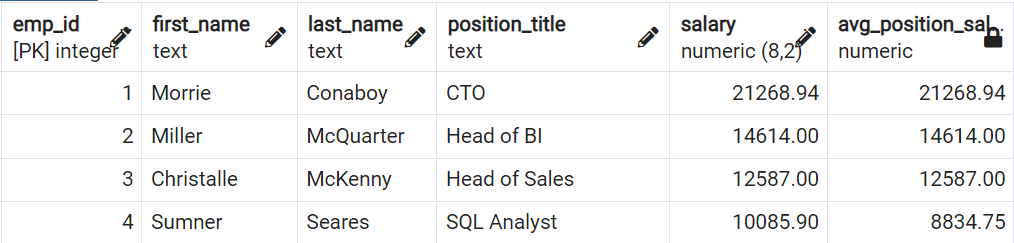
last\_name,

position\_title,

salary

and a column that returns the average salary for every job\_position.

Order the results by the emp\_id.



1. SELECT
2. emp\_id,
3. first\_name,
4. last\_name,
5. position\_title,
6. salary,
7. ROUND(AVG(salary) OVER(PARTITION BY position\_title),2) as avg\_position\_sal
8. FROM employees
9. ORDER BY 1;

**Task 8.2**

Difficulty: Advanced to Pro

How many people earn less than there avg\_position\_salary?

Write a query that answers that question.

Ideally, the output just shows that number directly.

*Answer:*

9

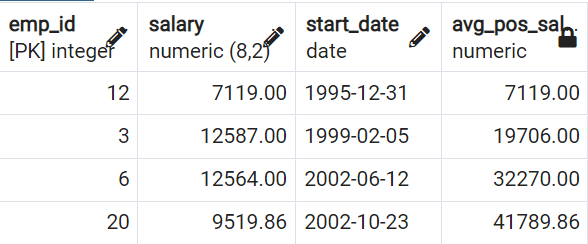
1. SELECT
2. COUNT(\*)
3. FROM (
4. SELECT
5. emp\_id,
6. salary,
7. ROUND(AVG(salary) OVER(PARTITION BY position\_title),2) as avg\_pos\_sal
8. FROM employees) a
9. WHERE salary<avg\_pos\_sal;

**Task 9:**

Difficulty: Advanced

Write a query that returns a running total of the salary development ordered by the start\_date.

In your calculation, you can assume their salary has not changed over time, and you can disregard the fact that people have left the company (write the query as if they were still working for the company).



*Question:*

What was the total salary after 2018-12-31?

*Answer:*

180202.70

1. SELECT
2. emp\_id,
3. salary,
4. start\_date,
5. SUM(salary) OVER(ORDER BY start\_date) as salary\_totals
6. FROM employees;

**Task 10:**

Difficulty: Pro / Very difficult

Create the same running total but now also consider the fact that people were leaving the company.

*Note:*

This challenge is actually very difficult.

Don't worry if you can't solve it you are not expected to do so.

It is possible to solve the challenge even without the hints.

If you want you can try to solve it using the hints and it is still a difficult challenge.

*Question:*

What was the total salary after 2018-12-31?

*Answer:*

166802.84

*Hint 1:*Use the view v\_employees\_info.

*Hint 2:*Create two separate queries: one with all employees and one with the people that have already left

*Hint 3:*In the first query use start\_date and in the second query use end\_date instead of the start\_date

*Hint 4:*Multiply the salary of the second query with (-1).

*Hint 5:*Create a subquery from that UNION and use a window function in that to create the running total.

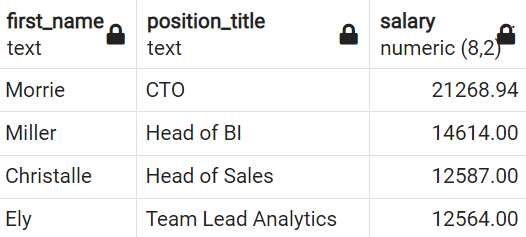
1. SELECT
2. start\_date,
3. SUM(salary) OVER(ORDER BY start\_date)
4. FROM (
5. SELECT
6. emp\_id,
7. salary,
8. start\_date
9. FROM employees
10. UNION
11. SELECT
12. emp\_id,
13. -salary,
14. end\_date
15. FROM v\_employees\_info
16. WHERE is\_active ='false'
17. ORDER BY start\_date) a

**Task 11**

Difficulty: Advanced to Pro

**Task 11.1**

Write a query that outputs only the top earner per position\_title including first\_name and position\_title and their salary.



*Question:*

What is the top earner with the position\_title SQL Analyst?

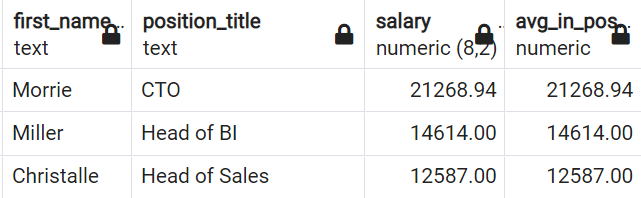
*Answer:*

Sumner with 10085.90

1. SELECT
2. first\_name,
3. position\_title,
4. salary
5. FROM employees e1
6. WHERE salary = (SELECT MAX(salary)
7. FROM employees e2
8. WHERE e1.position\_title=e2.position\_title)

**Task 11.2**

Add also the average salary per position\_title.



1. SELECT
2. first\_name,
3. position\_title,
4. salary,
5. (SELECT ROUND(AVG(salary),2) as avg\_in\_pos FROM employees e3
6. WHERE e1.position\_title=e3.position\_title)
7. FROM employees e1
8. WHERE salary = (SELECT MAX(salary)
9. FROM employees e2
10. WHERE e1.position\_title=e2.position\_title)

**Task 11.3**

Remove those employees from the output of the previous query that has the same salary as the average of their position\_title.

These are the people that are the only ones with their position\_title.

1. SELECT
2. first\_name,
3. position\_title,
4. salary,
5. (SELECT ROUND(AVG(salary),2) as avg\_in\_pos FROM employees e3
6. WHERE e1.position\_title=e3.position\_title)
7. FROM employees e1
8. WHERE salary = (SELECT MAX(salary)
9. FROM employees e2
10. WHERE e1.position\_title=e2.position\_title)
11. AND salary<>(SELECT ROUND(AVG(salary),2) as avg\_in\_pos FROM employees e3
12. WHERE e1.position\_title=e3.position\_title)

**Task 12**

Difficulty: Pro

Write a query that returns all meaningful aggregations of

- sum of salary,

- number of employees,

- average salary

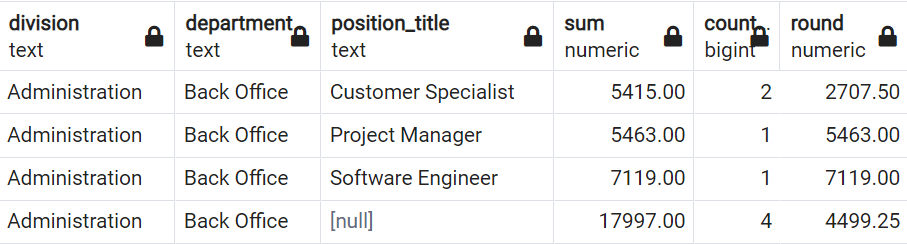
grouped by all meaningful combinations of

- division,

- department,

- position\_title.

Consider the levels of hierarchies in a meaningful way.



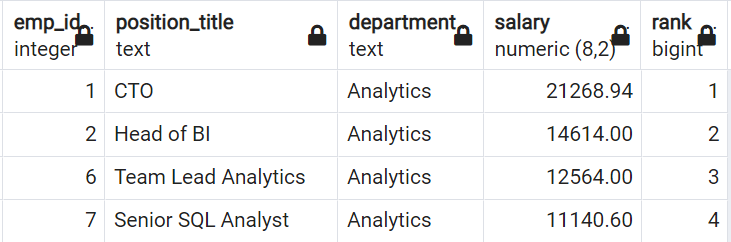
1. SELECT
2. division,
3. department,
4. position\_title,
5. SUM(salary),
6. COUNT(\*),
7. ROUND(AVG(salary),2)
8. FROM employees
9. NATURAL JOIN departments
10. GROUP BY
11. ROLLUP(
12. division,
13. department,
14. position\_title
15. )
16. ORDER BY 1,2,3

**Task 13**

Difficulty: Advanced to Pro

Write a query that returns all employees (emp\_id) including their position\_title, department, their salary, and the rank of that salary partitioned by department.

The highest salary per division should have rank 1.



*Question:*

Which employee (emp\_id) is in rank 7 in the department Analytics?

*Answer:*

emp\_id 26

1. SELECT
2. emp\_id,
3. position\_title,
4. department,
5. salary,
6. RANK() OVER(PARTITION BY department ORDER BY salary DESC)
7. FROM employees
8. NATURAL LEFT JOIN departments

**Task 14**

Difficulty: Pro

Write a query that returns only the top earner of each department including

their emp\_id, position\_title, department, and their salary.

*Question:*

Which employee (emp\_id) is the top earner in the department Finance?

*Answer:*

emp\_id 8

1. SELECT \* FROM
2. (
3. SELECT
4. emp\_id,
5. position\_title,
6. department,
7. salary,
8. RANK() OVER(PARTITION BY department ORDER BY salary DESC)
9. FROM employees
10. NATURAL LEFT JOIN departments) a
11. WHERE rank=1