Tech ABC Corp - HR Database

[Hazem Sayed - 12/23/2022]

Business Scenario

Business requirement

Tech ABC Corp saw explosive growth with a sudden appearance onto the gaming scene with their new AI-powered video game console. As a result, they have gone from a small 10 person operation to 200 employees and 5 locations in under a year. HR is having trouble keeping up with the growth, since they are still maintaining employee information in a spreadsheet. While that worked for ten employees, it has becoming increasingly cumbersome to manage as the company expands.

As such, the HR department has tasked you, as the new data architect, to design and build a database capable of managing their employee information.

Dataset

The HR dataset is an Excel workbook which consists of 206 records, with eleven columns. The data is in human readable format, and has not been normalized at all. The data lists the names of employees at Tech ABC Corp as well as information such as job title, department, manager's name, hire date, start date, end date, work location, and salary.

IT Department Best Practices

The IT Department has certain Best Practices policies for databases, as detailed in the Best Practices document.

Step 1

Data Architecture Foundations

Step 1: Data Architecture Foundations

Hi,

Welcome to Tech ABC Corp. We are excited to have some new talent onboard. As you may already know, Tech ABC Corp has recently experienced a lot of growth. Our AI powered video game console WOPR has been hugely successful and as a result, our company has grown from 10 employees to 200 in only 6 months (and we are projecting a 20% growth a year for the next 5 years). We have also grown from our Dallas, Texas office, to 4 other locations nationwide: New York City, NY, San Francisco, CA, Minneapolis, MN, and Nashville, TN.

While this growth is great, it is really starting to put a strain on our record keeping in HR. We currently maintain all employee information on a shared spreadsheet. When HR consisted of only myself, managing everyone on an Excel spreadsheet was simple, but now that it is a shared document I am having serious reservations about data integrity and data security. If the wrong person got their hands on the HR file, they would see the salaries of every employee in the company, all the way up to the president.

After speaking with Jacob Lauber, the manager of IT, he suggested I put in a request to have my HR Excel file converted into a database. He suggested I reach out to you as I am told you have experience in designing and building databases. When you are building this, please keep in mind that I want any employee with a domain login to be have read only access the database. I just don't want them having access to salary information. That needs to be restricted to HR and management level employees only. Management and HR employees should also be the only ones with write access. By our current estimates, 90% of users will be read only.

I also want to make sure you know that am looking to turn my spreadsheet into a live database, one I can input and edit information into. I am not really concerned with reporting capabilities at the moment. Since we are working with employee data we are required by federal regulations to maintain this data for at least 7 years; additionally, since this is considered business critical data, we need to make sure it gets backed up properly.

As a final consideration. We would like to be able to connect with the payroll department's system in the future. They maintain employee attendance and paid time off information. It would be nice if the two systems could interface in the future

I am looking forward to working with you and seeing what kind of database you design for us.

Thanks,
Sarah Collins
Head of HR

Data Architect Business Requirement

- **Purpose of the new database:**

  Sarah Collins; head of HR is requesting to build and design a new access-control restricted database for employees. That will help them in better data management, and to handle scaling issues with company growth.

- **Current data management solution:**

  Currently, the HR team is maintaining all employee information on a shared spreadsheet.

- **Describe current data available:**

Employee information is available in the current spreadsheet. That includes (but not limited to) employees personal information and salaries.

- **Additional data requests:**

  Integrate with other systems in the company as well as protect the data and ensure data integrity.

- **Who will own/manage data**

  HR department will own / manage the data in the database

- **Who will have access to database**
  - HR Team (Read and Write Access)
  - Management team (Read and Write Access for specific tables)
  - All Employees (Read Access)

Data Architect Business Requirement

- **Estimated size of database**

  Database that can contain up to 1000 rows.

- **Estimated annual growth**

  The company is projecting a 20% growth a year for the next 5 years.

- **Is any of the data sensitive/restricted**

  Salary information is restricted to the managerial level.

- **Backup Requirements**

  All database backup schedules should be set based on archived method since employees are updated once a month.

  - Archive: Backup schedule is a full backup 1x per month.

Data Architect Technical Requirement

- **Justification for the new database**
  1. Ensure Data Integrity and security.
  2. Data Retention for 7 years.
  3. Integrate with other systems.
  4. Weekly and Monthly Backups

- **Database objects**
  - Employees
  - Departments
  - Education Levels
  - Jobs
  - Salaries
  - Job Titles

- **Data ingestion**

  The initial data ingestion will occur via ETL. The HR team will interact with Web App connected to the database to add/update employees information.

Data Architect Technical Requirement

- **Data governance (Ownership and User access)**

  Ownership: HR Team will own and maintain the data.

  User Access: HR Team will see employees information, only managerial levels will be able to see jobs and salary information.

- **Scalability**

  Data replication and sharding be used to ensure scalability based on HR team needs.

- **Flexibility**
  - Ability to add new attributes to tables/entities
  - Ability to add new tables.
  - ETL is built to allow easily data loading in future to move data from staging table to respective tables.

- **Storage & retention**

  Storage (disk or in-memory): Databases are stored on spinning disk to reduce costs and allow better flexibility for storage increase.

  Retention: Data will be retained for 7 years.

- **Backup**

  All database backup schedules should be set based on archived method since employees information are updated once a month.

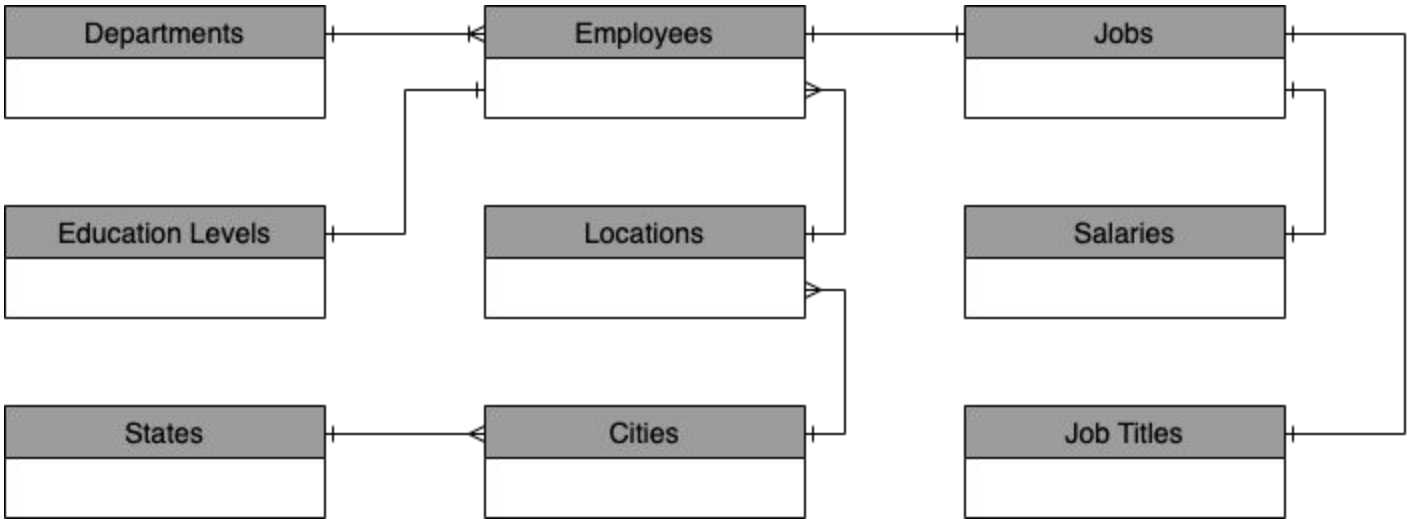
  - Archive: Backup schedule is a full backup 1x per month.

Step 2

Relational Database Design

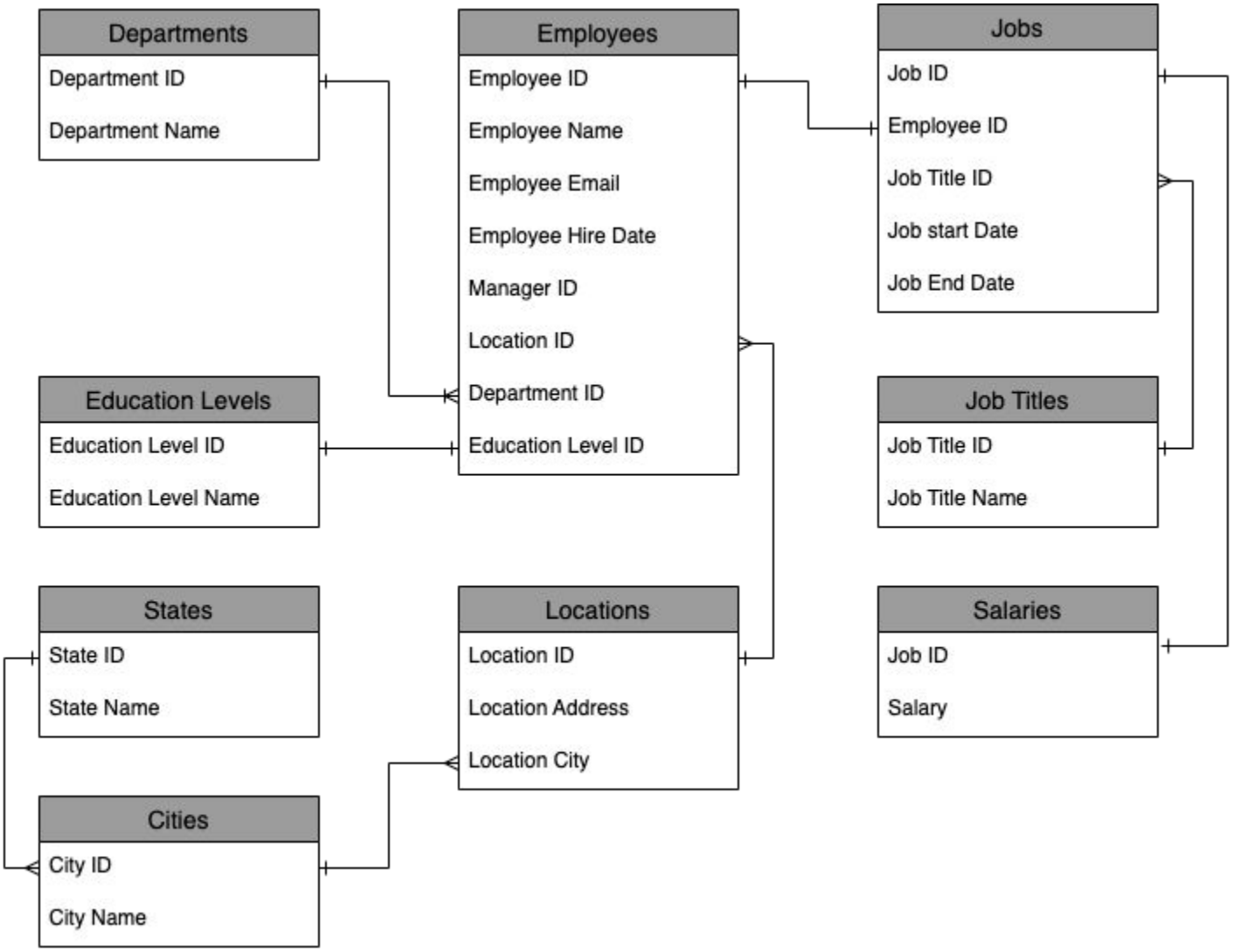
ERD

- **Conceptual**



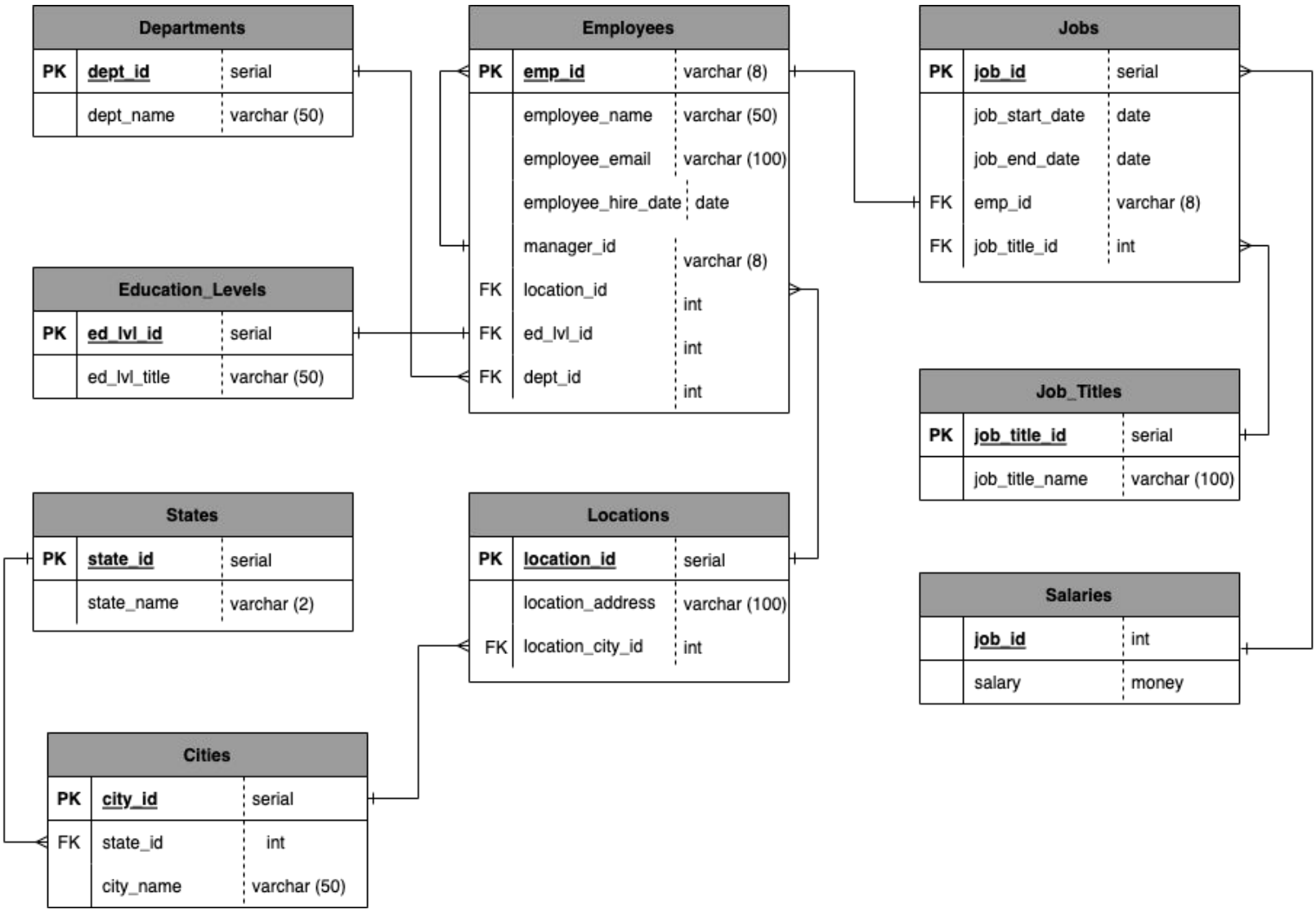
ERD

- **Logical**


Departments
Department ID
Department Name
Employees
Employee ID
Employee Name
Employee Email
Employee Hire Date
Manager ID
Location ID
Department ID
Education Level ID
Jobs
Job ID
Employee ID
Job Title ID
Job start Date
Job End Date
Education Levels
Education Level ID
Education Level Name
Job Titles
Job Title ID
Job Title Name
States
State ID
State Name
Locations
Location ID
Location Address
Location City
Salaries
Job ID
Salary
Cities
City ID
City Name

ERD

- **Physical**



Step 3

Create A Physical Database

DDL

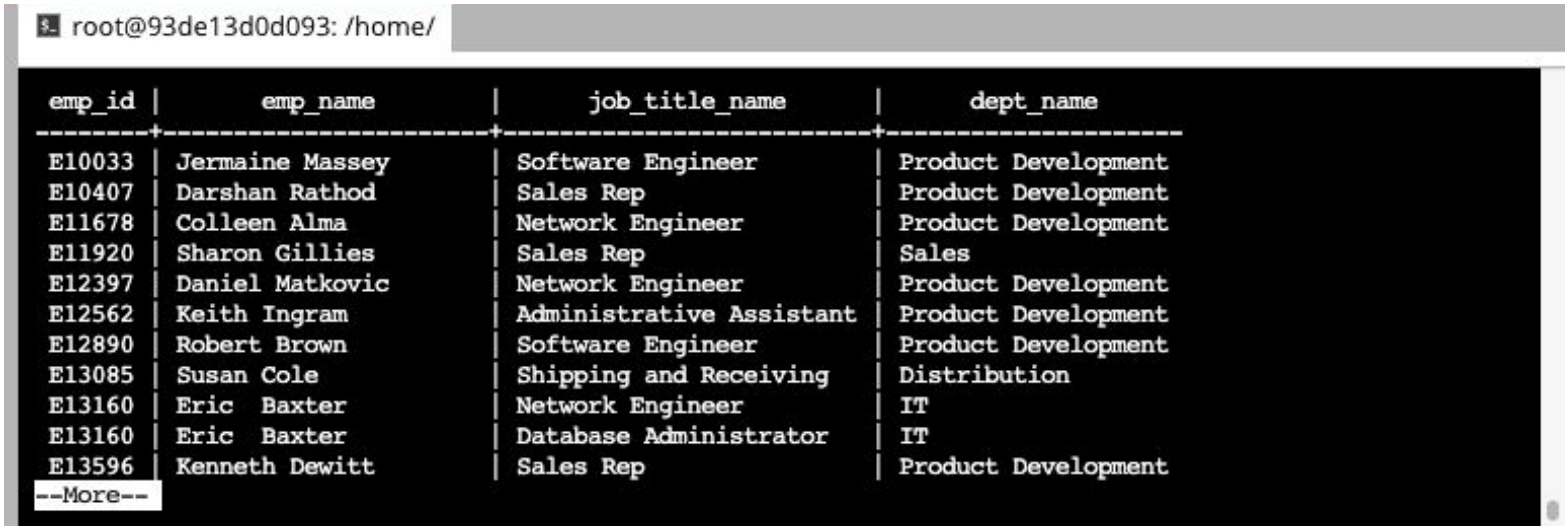
DDL SQL script capable of building the database designed in Step 2

```sql
CREATE TABLE Education_Levels (
    ed_lvl_id serial,
    ed_lvl_title varchar (50),
    PRIMARY KEY (ed_lvl_id)
);

CREATE TABLE Departments (
    dept_id serial,
    dept_name varchar (50),
    PRIMARY KEY(dept_id)
);

CREATE TABLE Job_Titles (
    job_title_id serial,
    job_title_name varchar(50),
    PRIMARY KEY (job_title_id)
);

CREATE TABLE States (
    state_id serial,
    state_name varchar (2),
    PRIMARY KEY (state_id)
);

CREATE TABLE Cities (
    city_id serial,
    city_name varchar (50),
    PRIMARY KEY (city_id),
    state_id int REFERENCES States (state_id)
);
```

```sql
CREATE TABLE Locations (
    location_id serial,
    location_address varchar (100),
    PRIMARY KEY (location_id),
    location_city_id int REFERENCES Cities (city_id)
);

CREATE TABLE Employees (
    emp_id varchar(8),
    emp_name varchar(50),
    emp_email varchar(100),
    emp_hire_dt date,
    PRIMARY KEY (emp_id),
    dept_id int REFERENCES Departments(dept_id),
    ed_lvl_id int REFERENCES Education_Levels (ed_lvl_id),
    location_id int REFERENCES Locations (location_id),
    manager_id varchar (8)
);

CREATE TABLE Jobs (
    job_id serial,
    job_start_dt date,
    job_end_dt date,
    PRIMARY KEY (job_id),
    emp_id varchar(8) REFERENCES Employees(emp_id),
    job_title_id int REFERENCES Job_Titles (job_title_id)
);

CREATE TABLE Salaries (
    job_id int REFERENCES Jobs(job_id),
    salary money
);
```

CRUD

- **Question 1: Return a list of employees with Job Titles and Department Names**



root@93de13d0d093: /home/

emp_id	emp_name	job_title_name	dept_name
E10033	Jermaine Massey	Software Engineer	Product Development
E10407	Darshan Rathod	Sales Rep	Product Development
E11678	Colleen Alma	Network Engineer	Product Development
E11920	Sharon Gillies	Sales Rep	Sales
E12397	Daniel Matkovic	Network Engineer	Product Development
E12562	Keith Ingram	Administrative Assistant	Product Development
E12890	Robert Brown	Software Engineer	Product Development
E13085	Susan Cole	Shipping and Receiving	Distribution
E13160	Eric Baxter	Network Engineer	IT
E13160	Eric Baxter	Database Administrator	IT
E13596	Kenneth Dewitt	Sales Rep	Product Development

--More--

```
SELECT
   EMP.emp_id,
   EMP.emp_name,
   JT.job_title_name,
   DEPT.dept_name
FROM
   Employees EMP
   JOIN Departments DEPT ON DEPT.dept_id = EMP.dept_id
   JOIN Jobs J ON J.emp_id = EMP.emp_id
   JOIN Job_Titles JT ON JT.job_title_id = J.job_title_id
```

CRUD

- **Question 2: Insert Web Programmer as a new job title**

```
postgres=# INSERT INTO Job_Titles (job_title_name) VALUES ('Web Programmer');
INSERT 0 1
postgres=#
```

```
INSERT INTO Job_Titles (job_title_name) VALUES ('Web Programmer')
```

CRUD

- **Question 3: Correct the job title from web programmer to web developer**


```
UPDATE
   Job_Titles
SET
   job_title_name = 'web developer'
WHERE
   job_title_name = 'Web Programmer'
```

CRUD

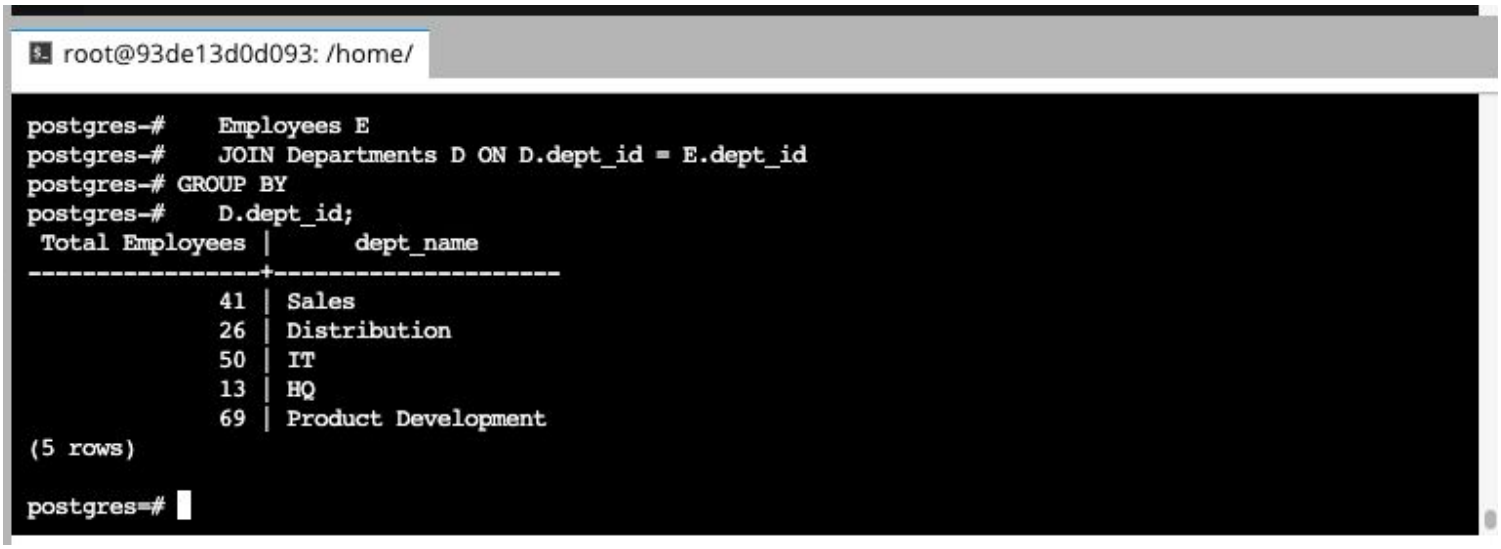
- **Question 4: Delete the job title Web Developer from the database**

```
postgres=# DELETE FROM
postgres-#     Job_Titles
postgres-# WHERE
postgres-#     job_title_name = 'web developer';
DELETE 1
postgres=#
```

```
DELETE FROM
   Job_Titles
WHERE
   job_title_name = 'web developer'
```

CRUD

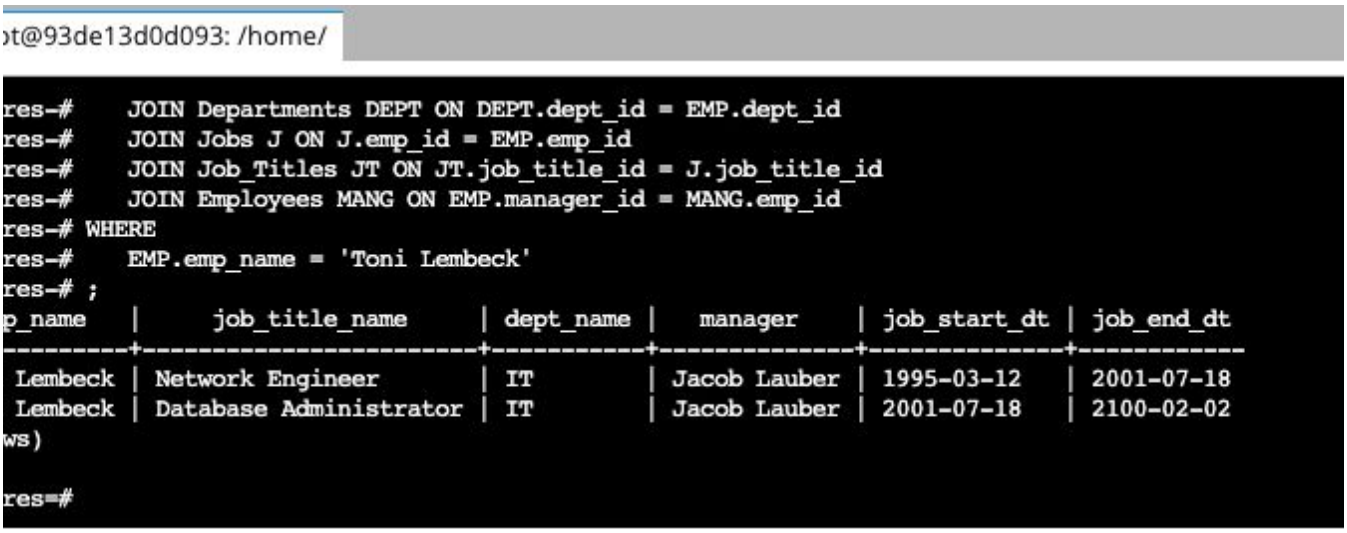
- **Question 5: How many employees are in each department?**




```
SELECT
   COUNT(EMP_ID) "Total Employees",
   D.dept_name
from
   Employees E
   JOIN Departments D ON D.dept_id = E.dept_id
GROUP BY
   D.dept_id;
```

CRUD

- **Question 6: Write a query that returns current and past jobs (include employee name, job title, department, manager name, start and end date for position) for employee Toni Lembeck.**



```
ot@93de13d0d093: /home/

res-#    JOIN Departments DEPT ON DEPT.dept_id = EMP.dept_id
res-#    JOIN Jobs J ON J.emp_id = EMP.emp_id
res-#    JOIN Job_Titles JT ON JT.job_title_id = J.job_title_id
res-#    JOIN Employees MANG ON EMP.manager_id = MANG.emp_id
res-# WHERE
res-#    EMP.emp_name = 'Toni Lembeck'
res-# ;
p_name   |     job_title_name     | dept_name |   manager    | job_start_dt | job_end_dt
---------+------------------------+-----------+--------------+--------------+------------
 Lembeck | Network Engineer       | IT        | Jacob Lauber | 1995-03-12   | 2001-07-18
 Lembeck | Database Administrator | IT        | Jacob Lauber | 2001-07-18   | 2100-02-02
ws)

res-#
```

```
SELECT
   EMP.emp_name,
   JT.job_title_name,
   DEPT.dept_name,
   MANG.emp_name as Manager,
   J.job_start_dt,
   J.job_end_dt
FROM
   Employees EMP
   JOIN Departments DEPT ON DEPT.dept_id = EMP.dept_id
   JOIN Jobs J ON J.emp_id = EMP.emp_id
   JOIN Job_Titles JT ON JT.job_title_id = J.job_title_id
   JOIN Employees MANG ON EMP.manager_id = MANG.emp_id
WHERE
   EMP.emp_name = 'Toni Lembeck'
```

CRUD

- **Question 7: Describe how you would apply table security to restrict access to employee salaries using an SQL server.**

To restrict access to employees' salaries information , we have to create Postgress roles first.

- postgres-# CREATE ROLE hr_user_role;

Then we will apply user role on the jobs table

- postgres-# GRANT SELECT ON jobs IN salaries TO hr_user_role;

Finally, we create a PostgreSQL user and assign the newly created role to that user.

- postgres-# CREATE USER hr1 WITH PASSWORD 'user_password';
- postgres-# GRANT hr_user_role TO hr_user;