Tech ABC Corp - HR Database

[Papa Moryba Kouate – 22 June 2021]

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 you will be working with 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 you should follow, 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:** maintain data integrity and make data more secure
- **Describe current data management solution:**they put all the information inside an excel file
- **Describe current data available:** employee id, employee' s name, email, hire date, job title, salary, department, manager name, start date, end date, location, address, city, state, education level
- **Additional data requests:** they ask to maintain this data for at least 7 years. In addiction they would like to be able to connect with the payroll department's system in the future.
- **Who will own/manage data:** the management and the HR employees

- **Who will have access to database:** every employees with domain login can have read access to the the database but they must not have access to the salary information. Instead the management and the HR employees can have write and read access and they can also access to the salay information.

Data Architect Business Requirement

- **Estimated size of database:** 206 rows and 15 columns
- **Estimated annual growth:** 20% growth per year for the next 5 years
- **Is any of the data sensitive/restricted:** salary data are restricted for employees who are not manager or HR employees

Data Architect Technical Requirement

- **Justification for the new database:** integrity of data and security.
- **Database objects:**
    - **Table →** education_level, employee, employment, manager, location, department, job, salary
    - **View Table →** manager (created in order to support the creation of the the table employment)
- **Data ingestion:** ETL

Data Architect Technical Requirement

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

  Ownership: HR Employees
  User Access: every employees. But the access is denied for salary information except for the management and the hr employees

- **Scalability:** replication

- **Flexibility:** a direct feed could be very useful in the future in order to connect the actual db with the payroll system

- **Storage & retention**

  Storage (disk or in-memory): disk
  Retention: 7 years

- **Backup:** full back-up weekly, with daily interval backups

Step 2

Relational Database Design

Step 2: Relational Database Design

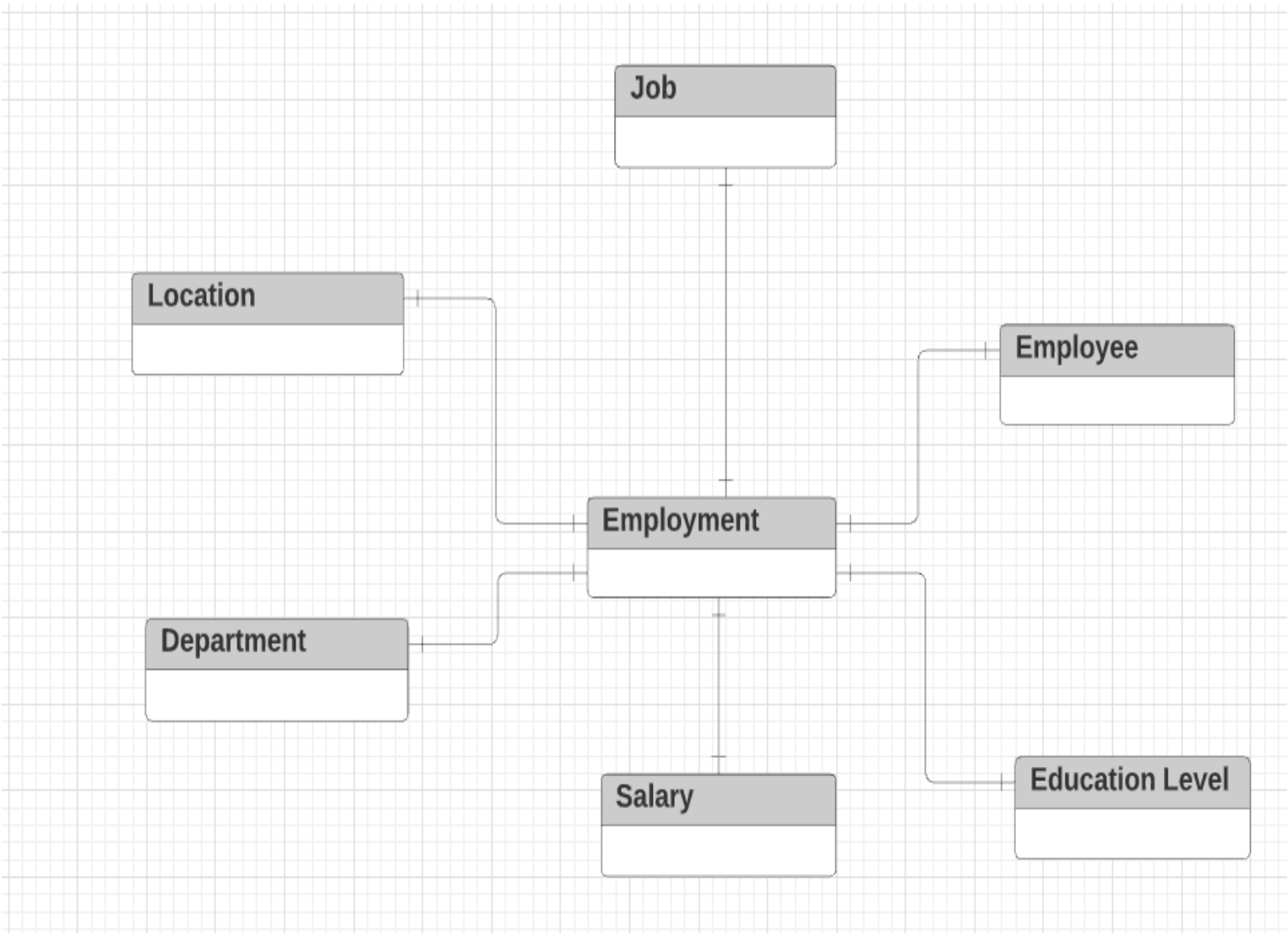
This step is where you will go through the process of designing a new database for Tech ABC Corp's HR department. Using the dataset provided, along with the requirements gathered in step one, you are going to develop a relational database set to the 3NF.

Using Lucidchart, you will create 3 entity relationship diagrams (ERDs) to show how you developed the final design for your data.

You will submit a screenshot for each of the 3 ERDs you create. You will find detailed instructions for developing each of the ERDs over the next several pages.

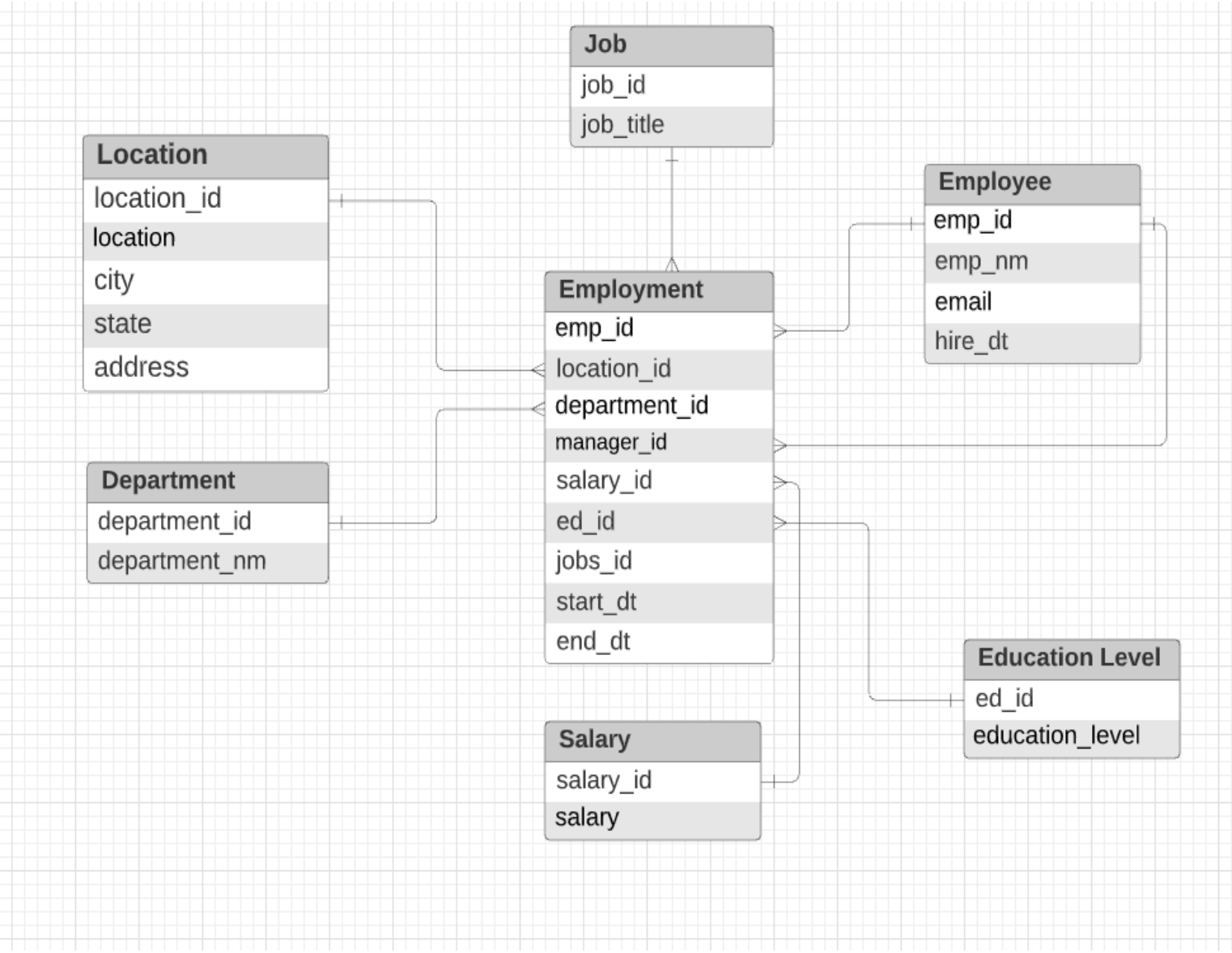
ERD

- **Conceptual**



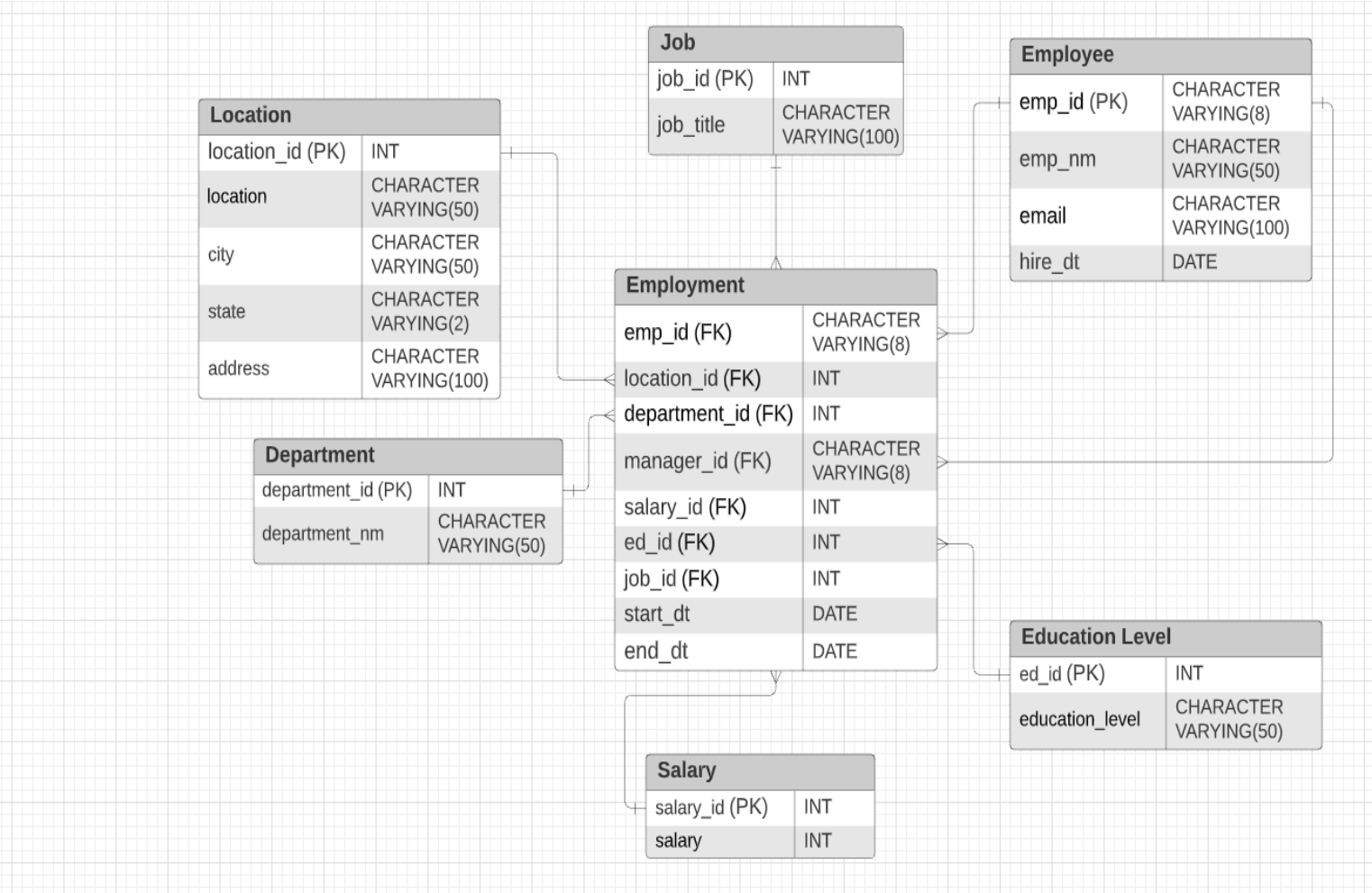
ERD

- **Logical**


Job
job_id
job_title
Location
location_id
location
city
state
address
Employee
emp_id
emp_nm
email
hire_dt
Employment
emp_id
location_id
department_id
manager_id
salary_id
ed_id
jobs_id
start_dt
end_dt
Department
department_id
department_nm
Education Level
ed_id
education_level
Salary
salary_id
salary

ERD

- **Physical**



Step 3

Create A Physical Database

Step 3: Create A Physical Database

In this step, you will be turning your database model into a physical database.

You will:

- Create the database using SQL DDL commands
- Load the data into your database, utilizing flat file ETL
- Answer a series of questions using CRUD SQL commands to demonstrate your database was created and populated correctly

Submission
For this step, you will need to submit SQL files containing all DDL SQL scripts used to create the database.

You will also have to submit screenshots showing CRUD commands, along with results for each of the questions found in the starter template.

Hints
Your DDL script will be graded by running the code you submit. Please ensure your SQL code runs properly!

Foreign keys cannot be created on tables that do not exist yet, so it may be easier to create all tables in the database, then to go back and run modify statements on the tables to create foreign key constraints.

After running CRUD commands like update, insert, or delete, run a `SELECT*` command on the affected table, so the reviewer can see the results of the command.

DDL

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

```
CREATE TABLE Employee (
    emp_id CHARACTER VARYING(8) PRIMARY KEY,
    emp_nm CHARACTER VARYING(50),
    email CHARACTER VARYING(100),
    hire_dt DATE);

CREATE TABLE Job (
    job_id SERIAL PRIMARY KEY,
    job_title CHARACTER VARYING(100));

CREATE TABLE Department (
    department_id SERIAL PRIMARY KEY,
    department_nm CHARACTER VARYING(50));

CREATE TABLE Salary (
    salary_id SERIAL PRIMARY KEY,
    salary INTEGER);

CREATE TABLE Location (
    location_id SERIAL PRIMARY KEY,
    location CHARACTER VARYING(50),
    state CHARACTER VARYING(2),
    city CHARACTER VARYING(50),
    address CHARACTER VARYING(100));

CREATE TABLE education_level (
    ed_id SERIAL PRIMARY KEY,
    education_level CHARACTER VARYING(50));
```

DDL

```
CREATE TABLE Employment (
    emp_id CHARACTER VARYING(8),
    location_id INTEGER,
    department_id INTEGER,
    salary_id INTEGER,
    ed_id INTEGER,
    job_id INTEGER,
    manager_id CHARACTER VARYING(8),
    start_dt DATE,
    end_dt DATE);
```

```
CREATE VIEW manager
AS SELECT s.emp_id AS manager_id,
p.manager AS manager_name
FROM proj_stg AS p
FULL JOIN (SELECT DISTINCT emp_id, emp_nm FROM proj_stg
WHERE emp_nm IN (SELECT DISTINCT manager FROM proj_stg)) AS s
ON p.manager=s.emp_nm;
```

CRUD

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

```
project_udacity=# SELECT e.emp_id, j.job_title, d.department_nm
project_udacity-# FROM employee AS e
project_udacity-# JOIN employment AS f
project_udacity-# ON e.emp_id = f.emp_id
project_udacity-# JOIN job AS j
project_udacity-# ON j.job_id = f.job_id
project_udacity-# JOIN department AS d
project_udacity-# ON d.department_id = f.department_id;
 emp_id |         job_title          |     department_nm
--------+----------------------------+-----------------------
 E21348 | Software Engineer          | Product Development
 E48884 | Network Engineer           | IT
 E32359 | Sales Rep                  | Sales
 E37523 | Administrative Assistant   | Product Development
 E93715 | Sales Rep                  | Sales
 E63041 | Manager                    | Distribution
 E15292 | Shipping and Receiving     | Distribution
 E94358 | Administrative Assistant   | Product Development
 E96966 | Software Engineer          | IT
 E14737 | Shipping and Receiving     | Distribution
 E50012 | Administrative Assistant   | IT
 E56444 | Sales Rep                  | Sales
 E10407 | Sales Rep                  | Product Development
 E93871 | Sales Rep                  | Product Development
 E44136 | Sales Rep                  | Sales
 E65052 | Sales Rep                  | Sales
 E34748 | Network Engineer           | IT
 E60752 | Design Engineer            | Product Development
 E34496 | Administrative Assistant   | Sales
 E22197 | Network Engineer           | IT
 E16346 | Administrative Assistant   | Product Development
 E88667 | Manager                    | Sales
 E37246 | Database Administrator     | IT
 E30317 | Network Engineer           | Product Development
 E42061 | Sales Rep                  | Sales
 E42522 | Design Engineer            | Product Development
 E45405 | Shipping and Receiving     | Distribution
 E38634 | Legal Counsel              | HQ
 E24539 | Sales Rep                  | Sales
 E25662 | Sales Rep                  | Product Development
 E12890 | Software Engineer          | Product Development
 E15292 | Software Engineer          | IT
 E21696 | Legal Counsel              | Product Development
 E75081 | Legal Counsel              | HQ
 E91182 | Sales Rep                  | Product Development
 E12562 | Administrative Assistant   | Product Development
 E71792 | Legal Counsel              | Sales
 E29652 | Software Engineer          | Product Development
 E43694 | Legal Counsel              | Sales
 E36988 | Shipping and Receiving     | Distribution
 E25640 | Administrative Assistant   | HQ
 E42681 | Sales Rep                  | Product Development
 E17054 | President                  | HQ
 E35856 | Software Engineer          | IT
 E67190 | Sales Rep                  | Product Development
```

CRUD

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

```
project_udacity=# INSERT INTO job(job_title) VALUES ('Web Programmer');
INSERT 0 1
project_udacity=# SELECT * FROM job;
 job_id |        job_title
--------+--------------------------
      1 | Shipping and Receiving
      2 | Sales Rep
      3 | Administrative Assistant
      4 | Design Engineer
      5 | Database Administrator
      6 | Software Engineer
      7 | Manager
      8 | Legal Counsel
      9 | President
     10 | Network Engineer
     11 | Web Programmer
(11 rows)
```

CRUD

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

```
project_udacity=# UPDATE job SET job_title='Web Developer' WHERE job_title='Web Programmer';
UPDATE 1
project_udacity=# SELECT * FROM job;
 job_id |        job_title
--------+-------------------------
      1 | Shipping and Receiving
      2 | Sales Rep
      3 | Administrative Assistant
      4 | Design Engineer
      5 | Database Administrator
      6 | Software Engineer
      7 | Manager
      8 | Legal Counsel
      9 | President
     10 | Network Engineer
     11 | Web Developer
(11 rows)
```

CRUD

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

```
project_udacity=# DELETE FROM job WHERE job_title='Web Developer';
DELETE 1
project_udacity=# SELECT * FROM job;
 job_id |         job_title
--------+---------------------------
      1 | Shipping and Receiving
      2 | Sales Rep
      3 | Administrative Assistant
      4 | Design Engineer
      5 | Database Administrator
      6 | Software Engineer
      7 | Manager
      8 | Legal Counsel
      9 | President
     10 | Network Engineer
(10 rows)
```

CRUD

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

```
project_udacity=# SELECT d.department_nm, COUNT(e.emp_id)
project_udacity-# FROM department AS d
project_udacity-# JOIN employment AS f
project_udacity-# ON d.department_id = f.department_id
project_udacity-# JOIN employee AS e
project_udacity-# ON e.emp_id = f.emp_id
project_udacity-# GROUP BY d.department_nm;
    department_nm    | count
---------------------+-------
 Product Development |    70
 HQ                  |    13
 Distribution        |    27
 Sales               |    41
 IT                  |    54
(5 rows)
```

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.**

```
project_udacity=# WITH sub AS (SELECT DISTINCT z.emp_id AS manager_id, z.emp_nm AS manager
project_udacity(# FROM employee AS z
project_udacity(# JOIN employment AS w
project_udacity(# ON z.emp_id = w.manager_id)
project_udacity-#
project_udacity-# SELECT DISTINCT e.emp_nm, j.job_title, d.department_nm, s.manager, f.start_dt, f.end_dt
project_udacity-# FROM employee AS e
project_udacity-# JOIN employment AS f
project_udacity-# ON e.emp_id = f.emp_id
project_udacity-# JOIN department AS d
project_udacity-# ON d.department_id = f.department_id
project_udacity-# JOIN sub AS s
project_udacity-# ON s.manager_id = f.manager_id
project_udacity-# JOIN job AS j
project_udacity-# ON j.job_id = f.job_id
project_udacity-# WHERE e.emp_nm = 'Toni Lembeck';
    emp_nm    |       job_title        | department_nm |   manager    |  start_dt  |   end_dt
--------------+------------------------+---------------+--------------+------------+------------
 Toni Lembeck | Database Administrator | IT            | Jacob Lauber | 2001-07-18 | 2100-02-02
 Toni Lembeck | Network Engineer       | IT            | Jacob Lauber | 1995-03-12 | 2001-07-18
(2 rows)
```

CRUD

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

  In order to restrict access to employee salaries, I think the best way is to apply **row-level security** that permits to grant the access to the salary table only to the management and HR employees.

Step 4

Above and Beyond

(optional)

Step 4: Above and Beyond

This last step is called Above and Beyond. In this step, I have proposed 3 challenges for you to complete, which are above and beyond the scope of the project. This is a chance to flex your coding muscles and show everyone how good you really are.

These challenge steps will bring your project even more in line with a real-world project, as these are the kind of "finishing touches" that will make your database more usable. Imagine building a car without air conditioning or turn signals. Sure, it will work, but who would want to drive it.

I encourage you to take on these challenges in this course and any future courses you take. I designed these challenges to be a challenge to your current abilities, but I ensured they are not an unattainable challenge. Remember, these challenges are completely optional - you can pass the project by doing none of them, or just some of them, but I encourage you to at least attempt them!

Standout Suggestion 1

Create a view that returns all employee attributes; results should resemble initial Excel file

```
project_udacity=# CREATE VIEW start_file AS SELECT e.emp_id,
project_udacity-# e.emp_nm,
project_udacity-# e.email,
project_udacity-# e.hire_dt,
project_udacity-# j.job_title,
project_udacity-# s.salary,
project_udacity-# d.department_nm,
project_udacity-# sub.manager,
project_udacity-# f.start_dt,
project_udacity-# f.end_dt,
project_udacity-# l.location,
project_udacity-# l.address,
project_udacity-# l.city,
project_udacity-# l.state,
project_udacity-# x.education_level
project_udacity-# FROM employee AS e
project_udacity-# JOIN employment AS f
project_udacity-# ON e.emp_id = f.emp_id
project_udacity-# JOIN salary AS s
project_udacity-# ON s.salary_id = f.salary_id
project_udacity-# JOIN location AS l
project_udacity-# ON l.location_id = f.location_id
project_udacity-# JOIN (SELECT DISTINCT z.emp_id AS manager_id, z.emp_nm AS manager
project_udacity(# FROM employee AS z
project_udacity(# JOIN employment AS w
project_udacity(# ON z.emp_id = w.manager_id) AS sub
project_udacity-# ON sub.manager_id = f.manager_id
project_udacity-# JOIN job AS j
project_udacity-# ON j.job_id= j.job_id
project_udacity-# JOIN department AS d
project_udacity-# ON d.department_id=f.department_id
project_udacity-# JOIN education_level AS x
project_udacity-# ON x.ed_id = f.ed_id;
CREATE VIEW
project_udacity=# SELECT * FROM start_file;
 emp_id |       emp_nm       |              email              | hire_dt  |       job_title        | salary |    department_nm    |      manager       | start_dt | end_dt | location |      address       |    city
   | state |         education_level
--------+--------------------+---------------------------------+----------+------------------------+--------+---------------------+--------------------+----------+----------+----------+--------------------+-----------
---+-------+---------------------------------
 E21348 | Nital Thaker       | Nital.Thaker@TechCorp.com       | 2016-09-28 | Shipping and Receiving | 148313 | Product Development | Conner Kinch       | 2016-09-28 | 2100-07-11 | HQ       | 1 Tech ABC Corp Way | Dallas
   | TX    | Bachelors Degree
 E48884 | Stacey Lewis       | Stacey.Lewis@TechCorp.com       | 2008-06-01 | Shipping and Receiving | 103714 | IT                  | Jacob Lauber       | 2008-06-01 | 2100-03-19 | Midwest  | 1300 Nicollet Mall  | Minnapolis
   | MN    | Bachelors Degree
 E32359 | Jen  Frangias      | Jen .Frangias@TechCorp.com      | 2019-03-24 | Shipping and Receiving | 102779 | Sales               | Jennifer De La Garza | 2019-03-24 | 2100-05-12 | HQ       | 1 Tech ABC Corp Way | Dallas
   | TX    | Bachelors Degree
 E37523 | Alan Mecklet       | Alan.Mecklet@TechCorp.com       | 2017-12-02 | Shipping and Receiving |  35638 | Product Development | Conner Kinch       | 2017-12-02 | 2100-01-21 | East Coast | 165 Broadway        | New York Ci
ty | NY    | Bachelors Degree
 E93715 | Charles Barker     | Charles.Barker@TechCorp.com     | 1998-04-29 | Shipping and Receiving | 196650 | Sales               | Jennifer De La Garza | 1998-04-29 | 2100-06-04 | East Coast | 165 Broadway        | New York Ci
ty | NY    | Bachelors Degree
 E63041 | Allison Gentle     | Allison.Gentle@TechCorp.com     | 1995-08-22 | Shipping and Receiving | 176000 | Distribution        | Tyrone Hutchison   | 1995-08-22 | 2100-03-10 | West Coast | 705 James Way       | San Francis
co | CA    | Masters of Business Administration
 E15292 | Melinda Fisher     | Melinda.Fisher@TechCorp.com     | 2011-02-06 | Shipping and Receiving |  32933 | Distribution        | Allison Gentle     | 2007-02-22 | 2011-02-06 | HQ       | 1 Tech ABC Corp Way | Dallas
   | TX    | Bachelors Degree
 E94358 | Muhammed Rubel     | Muhammed.Rubel@TechCorp.com     | 2007-07-08 | Shipping and Receiving |  48239 | Product Development | Conner Kinch       | 2007-07-08 | 2100-01-27 | HQ       | 1 Tech ABC Corp Way | Dallas
   | TX    | Associates Degree
 E96966 | Lu Huang           | Lu.Huang@TechCorp.com           | 2014-12-19 | Shipping and Receiving | 134250 | IT                  | Jacob Lauber       | 2014-12-19 | 2100-07-06 | South    | 422 Broadway        | Nashville
   | TN    | Bachelors Degree
 E14737 | Juan Cosme         | Juan.Cosme@TechCorp.com         | 2012-07-22 | Shipping and Receiving |  26050 | Distribution        | Allison Gentle     | 2012-07-22 | 2100-06-07 | East Coast | 165 Broadway        | New York Ci
ty | NY    | Masters Degree
```

Standout Suggestion 2

Create a stored procedure with parameters that returns current and past jobs (include employee name, job title, department, manager name, start and end date for position) when given an employee name.

```
project_udacity=# CREATE PROCEDURE employee_data(name CHARACTER VARYING)
project_udacity-# AS $BODY$
project_udacity$# SELECT emp_nm, job_title, department_nm, manager, start_dt, end_dt
project_udacity$# FROM proj_stg
project_udacity$# WHERE emp_nm = name;
project_udacity$# $BODY$
project_udacity-# LANGUAGE SQL;
CREATE PROCEDURE
project_udacity=# CALL employee_data('Toni Lembeck');
CALL
```

Standout Suggestion 3

Implement user security on the restricted salary attribute.

I try to grant the privilege only for those table that don't contain salary amount.

```
CREATE USER NoMgr;
GRANT SELECT ON employee TO NoMgr;
GRANT SELECT ON area TO NoMgr;
GRANT SELECT ON residence TO NoMgr;
GRANT SELECT ON date TO NoMgr;
```

Appendix