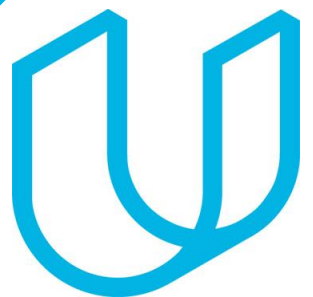


Tech ABC Corp - HR Database

[Juan Barbosa & 2022-05-24]



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 able to 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.

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.

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**

Because of the increase in size of the company, there are concerns about data integrity and data security of shared Excel spreadsheet.

- **Data to be stored**

The following data is to be stored in the database:

EMP_ID, EMP_NM, EMAIL
HIRE_DT, JOB_TITLE, SALARY
DEPARTMENT, MANAGER
START_DT, END_DT
LOCATION, ADDRESS, CITY
STATE, EDUCATION LEVEL

- **Who will own/manage data**

The department that owns the data is HR

- **Who will have access to database**

Employees: read only, salary data hidden

Managers: write, salary data not hidden

HR employees: write, salary data not hidden

Data Architect Business Requirement

- **Estimated size of database**

For the next 5 years the database is estimated to have about 500 rows

- **Estimated annual growth**

20 %

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

Salary data is to be restricted

- **Data retention and backup requirements**

Federal regulations require to maintain data for at least 7 years.

This is considered business critical data, thus, it should be backed up properly

Data Architect Technical Requirement

- **Justification for the new database**

Data integrity of an Excel file that has no control of the types used in each cell.

Scale of the data in the following years

- **Database objects**

salaries, employees, locations, managers, education_levels, job_titles, departments, locations

- **Data ingestion**

Since the HR needs to have access to writing data in the database, and we do not want to ask them for an API, or SQL statements, the best way is to use an ETL where HR writes the input in a flat file

Data Architect Technical Requirement

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

Ownership: HR department

User Access: Employees, Managers and HR employees. Access must be granted using specific data in database

- **Scalability & Flexibility considerations**

Replicas might be considered in order to handle the increasing reading operations

The database is to be build using 3NF to best handle new fields if required in the future

- **Storage & retention**

Storage (disk or in-memory): disk, no high level computations are required

Retention: for at least 7 years

- **Backup**

Standard backup schedules are enough, as data should not vary in a daily basis



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.

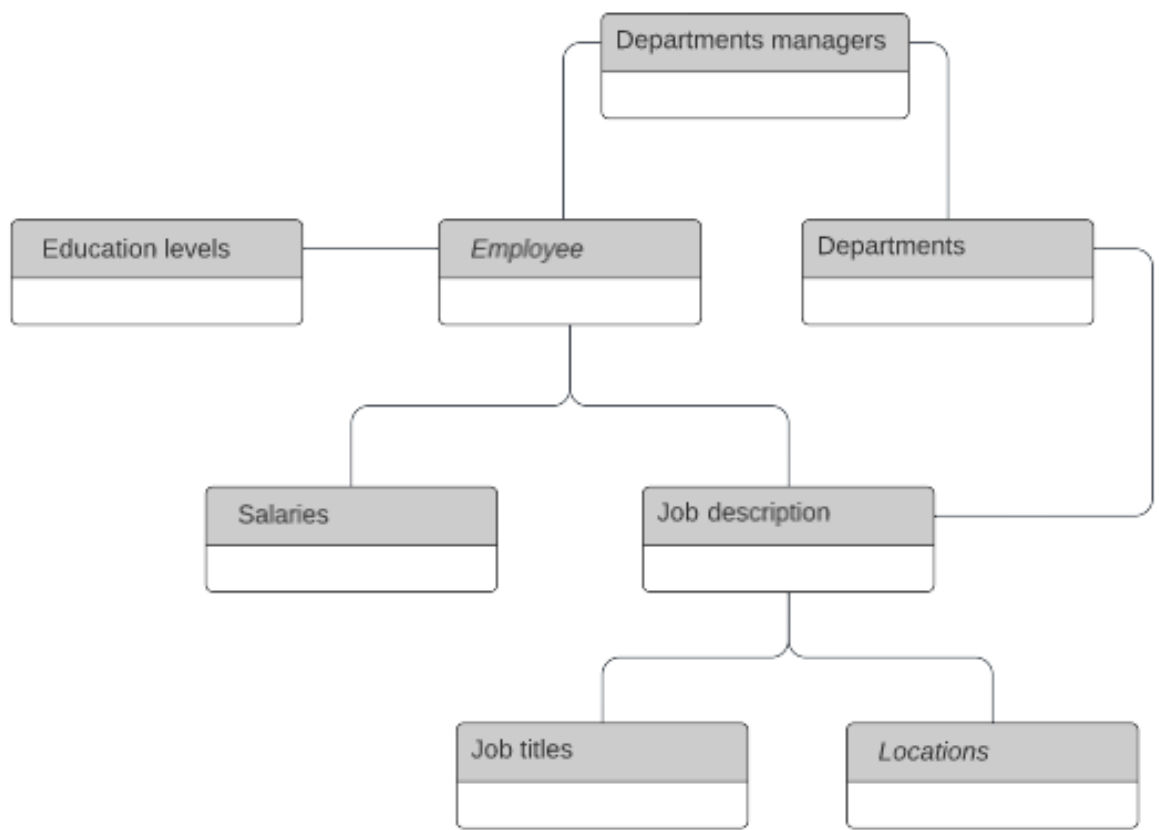
You will submit a screenshot for each of the 3 ERDs you create. You will find detailed instructions for developing each of the ERDs over the next several pages.

ERD

- **Conceptual**

This is the most general level of data modeling. At the conceptual level, you should be thinking about creating entities that represent business objects for the database. Think broadly here. Attributes (or column names) are not required at this point, but relationship lines are required (although Crow's foot notation is not needed at this level). Create at least three entities for this model; thinking about the 3NF will aid you in deciding the type of entities to create.

Use Lucidchart's built-in template for DBMS ER Diagram UML.

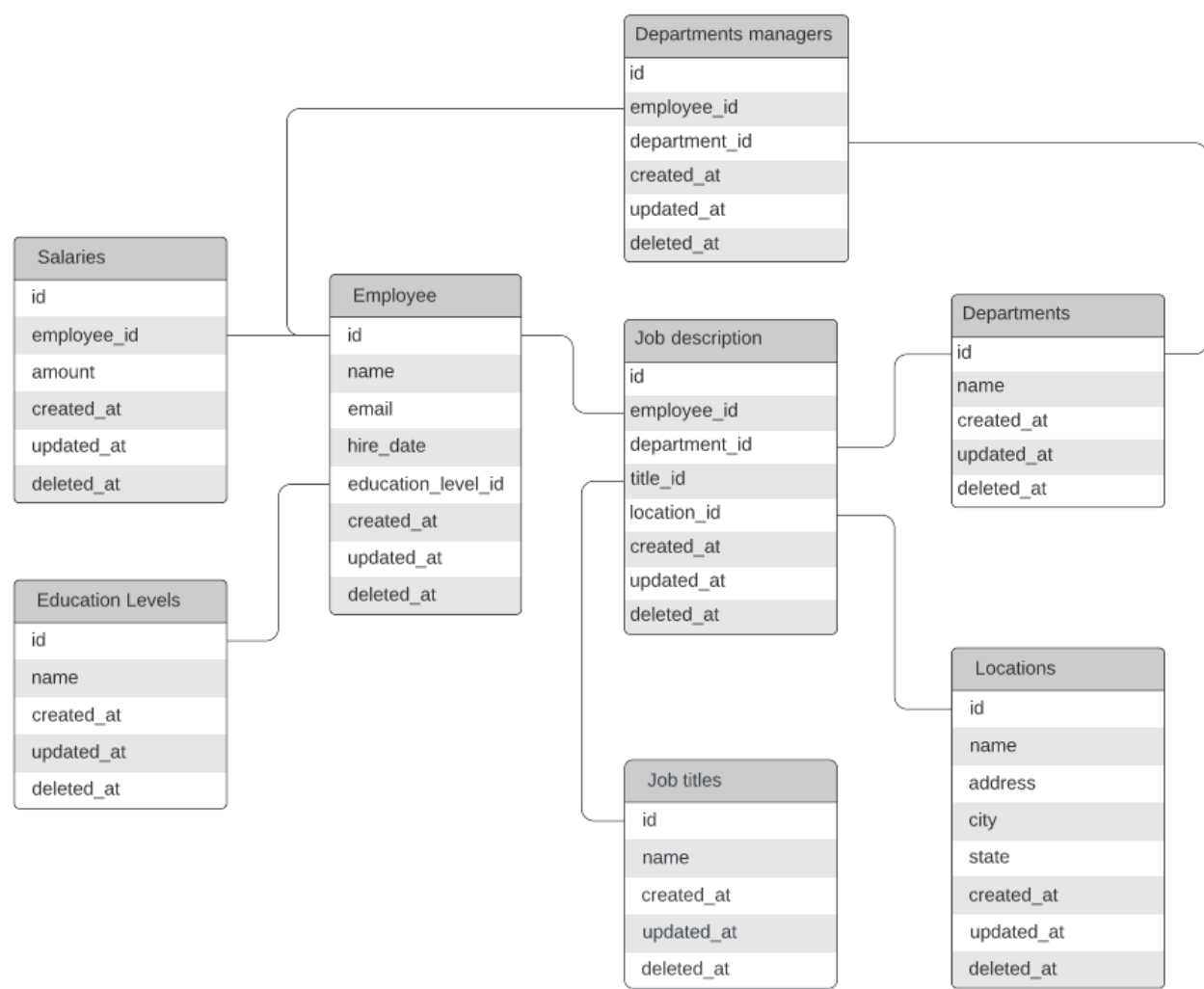


ERD

- **Logical**

The logical model is the next level of refinement from the conceptual ERD. At this point, you should have normalized the data to the 3NF. Attributes should also be listed now in the ERD. You can still use human-friendly entity and attribute names in the logical model, and while relationship lines are required, Crow's foot notation is still not needed at this point.

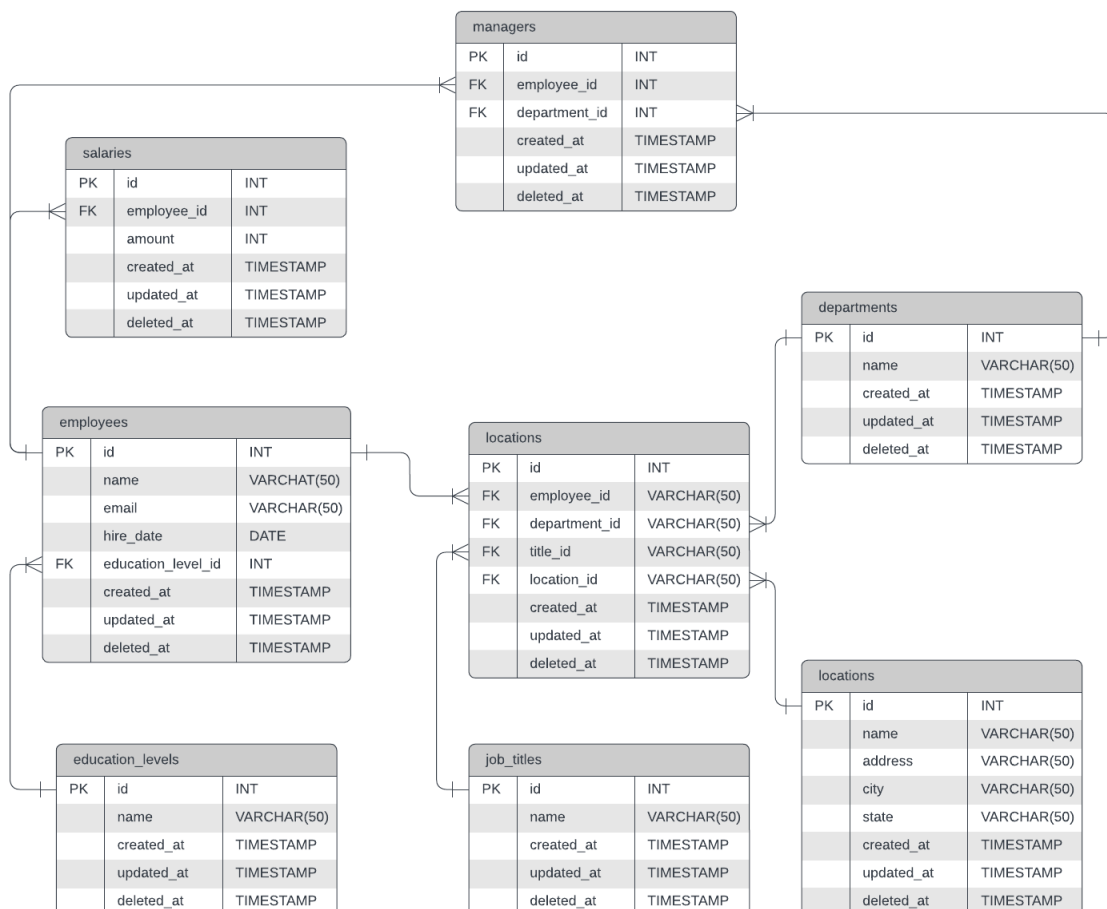
Use Lucidchart's built-in template for DBMS ER Diagram UML.



ERD

- Physical

The physical model is what will be built in the database. Each entity should represent a database table, complete with column names and data types. Primary keys and foreign keys should also be represented here. Primary keys should be in bold type with the (PK) designation following the field name. Foreign keys should be in normal type face, but have the designation (FK) after the column name. Finally, in the physical model, Crow's foot notation is important.





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

Hints

The 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.

```
1 CREATE TABLE IF NOT EXISTS education_levels (  
2     id INT NOT NULL,  
3     name VARCHAR(50),  
4     created_at TIMESTAMP DEFAULT NOW(),  
5     updated_at TIMESTAMP DEFAULT NOW(),  
6     deleted_at TIMESTAMP DEFAULT NULL,  
7     PRIMARY KEY (id));  
8  
9  
10 CREATE TABLE IF NOT EXISTS job_titles (  
11     id INT NOT NULL,  
12     name VARCHAR(50),  
13     created_at TIMESTAMP DEFAULT NOW(),  
14     updated_at TIMESTAMP DEFAULT NOW(),  
15     deleted_at TIMESTAMP DEFAULT NULL,  
16     PRIMARY KEY (id));  
17  
18  
19 CREATE TABLE IF NOT EXISTS locations (  
20     id INT NOT NULL,  
21     name VARCHAR(50),  
22     created_at TIMESTAMP DEFAULT NOW(),  
23     updated_at TIMESTAMP DEFAULT NOW(),  
24     deleted_at TIMESTAMP DEFAULT NULL,  
25     PRIMARY KEY (id));
```

Data output Messages Notifications

CREATE TABLE

Query returned successfully in 111 msec.

CRUD

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

Query

Query History

```
1 SELECT
2     employees.id,
3     employees.name,
4     job_titles.name AS job_title,
5     departments.name as department
6 FROM
7     employees
8     JOIN
9     job_descriptions ON employees.id = job_descriptions.employee_id
10    AND job_descriptions.deleted_at IS NULL
11    JOIN
12    job_titles ON job_titles.id = job_descriptions.title_id
13    JOIN
14    departments ON departments.id = job_descriptions.department_id
15 ORDER BY employees.id;
```

Data output

Messages

Notifications

≡+

	id character varying (50)	name character varying (50)	job_title character varying (50)	department character varying (50)
1	E10033	Jermaine Massey	Software Engineer	Product Development
2	E10407	Darshan Rathod	Sales Rep	Product Development
3	E11678	Colleen Alma	Network Engineer	Product Development
4	E11920	Sharon Gillies	Sales Rep	Sales
5	E12397	Daniel Matkovic	Network Engineer	Product Development
6	E12562	Keith Ingram	Administrative Assista...	Product Development
7	E12890	Robert Brown	Software Engineer	Product Development
8	E13085	Susan Cole	Shipping and Receiving	Distribution
9	E13160	Eric Baxter	Database Administrator	IT
10	E13596	Kenneth Dewitt	Sales Rep	Product Development
11	E14737	Juan Cosme	Shipping and Receiving	Distribution
12	E14913	Aaron Gordon	Network Engineer	Product Development

CRUD

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

```
1 INSERT INTO job_titles (id, name)
2 VALUES
3     (11, 'Web Programmer');
4
5 SELECT
6     *
7 FROM
8     job_titles;
```

Data output Messages Notifications



	id [PK] integer	name character varying (50)	created_at timestamp without time zone	updated_at timestamp without time zone	deleted_at timestamp without time zone
1	1	Administrative Assist...	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
2	2	Database Administrat...	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
3	3	Design Engineer	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
4	4	Legal Counsel	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
5	5	Manager	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
6	6	Network Engineer	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
7	7	President	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
8	8	Sales Rep	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
9	9	Shipping and Receiving	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
10	10	Software Engineer	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
11	11	Web Programmer	2022-05-30 18:57:40.447649	2022-05-30 18:57:40.447649	[null]

CRUD

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

Query








Query History






```
1 UPDATE job_titles
2 SET
3     name = 'Web Developer',
4     updated_at = NOW()
5 WHERE
6     id = 11;
7
8 SELECT
9     *
10 FROM
11     job_titles;
```

Data output

Messages

Notifications



	id [PK] integer 	name character varying (50) 	created_at timestamp without time zone 	updated_at timestamp without time zone 	deleted_at timestamp without time zone 
1	1	Administrative Assista...	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
2	2	Database Administrator	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
3	3	Design Engineer	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
4	4	Legal Counsel	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
5	5	Manager	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
6	6	Network Engineer	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
7	7	President	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
8	8	Sales Rep	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
9	9	Shipping and Receiving	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
10	10	Software Engineer	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
11	11	Web Developer	2022-05-30 18:57:40.447649	2022-05-30 18:58:12.1852	[null]

CRUD

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

Query Query History

```
1  -- UNCOMMENT FOR HARD DELETE
2
3  /*
4  DELETE FROM job_titles
5  WHERE
6      id = 11;
7  */
8
9  UPDATE job_titles
10 SET
11     deleted_at = NOW()
12 WHERE
13     id = 11;
14
15 SELECT
16     *
17 FROM
18     job_titles;
```

Data output Messages Notifications

	id [PK] integer	name character varying (50)	created_at timestamp without time zone	updated_at timestamp without time zone	deleted_at timestamp without time zone
1	1	Administrative Assista...	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
2	2	Database Administrator	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
3	3	Design Engineer	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
4	4	Legal Counsel	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
5	5	Manager	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
6	6	Network Engineer	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
7	7	President	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
8	8	Sales Rep	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
9	9	Shipping and Receiving	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
10	10	Software Engineer	2022-05-30 12:09:36.842	2022-05-30 12:09:36.842	[null]
11	11	Web Developer	2022-05-30 18:57:40.447649	2022-05-30 18:58:12.1852	2022-05-30 18:58:50.938016

CRUD

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

Query Query History

```
1 SELECT
2     departments.name AS department,
3     COUNT(*) AS employees
4 FROM
5     employees
6     JOIN
7     job_descriptions ON employees.id = job_descriptions.employee_id
8     AND job_descriptions.deleted_at IS NULL
9     JOIN
10    departments ON departments.id = job_descriptions.department_id
11 GROUP BY departments.name;
```

Data output Messages Notifications

	department character varying (50) 	employees bigint 
1	Product Development	69
2	HQ	13
3	Distribution	25
4	Sales	40
5	IT	52

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.**

Query Query History

```
3  job_titles.name AS job_title,  
4  departments.name AS department,  
5  DATE(job_descriptions.created_at) AS start_date,  
6  DATE(job_descriptions.deleted_at) AS end_date,  
7  manager.name as manager  
8  FROM  
9  employees  
10     JOIN  
11     job_descriptions ON employees.id = job_descriptions.employee_id  
12     JOIN  
13     departments ON departments.id = job_descriptions.department_id  
14     JOIN  
15     job_titles ON job_titles.id = job_descriptions.title_id  
16     JOIN  
17     managers ON managers.department_id = departments.id  
18     JOIN  
19     employees AS manager ON managers.employee_id = manager.id  
20  WHERE  
21     employees.name = 'Toni Lembeck';
```

Data output Messages Notifications

	name character varying (50)	job_title character varying (50)	department character varying (50)	start_date date	end_date date	manager character varying (50)
1	Toni Lembeck	Network Engineer	IT	1995-03-12	2001-07-17	Jacob Lauber
2	Toni Lembeck	Database Administrat...	IT	2001-07-18	[null]	Jacob Lauber

CRUD

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

If users are granted SELECT privileges on all tables, I would then REVOKE SELECT privileges on the *salaries* table for every employee not listed as manager or HR.

On the other hand if each SELECT is granted individually for each table in the database, then I would skip the *salaries* table for a regular employee.