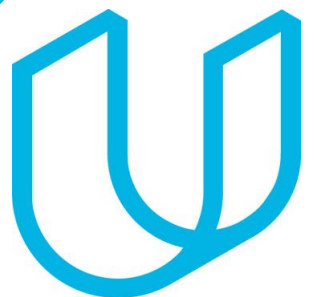


Tech ABC Corp - HR Database

Janis Puris, 2021-06-23



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 become 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:**

Online transactional processing datastore for Human Resources management (HRM).

- **Describe current data management solution:**

Currently the data is managed in a single, shared excel spreadsheet.

- **Describe current data available:**

Currently available data consists of flat structured data. The data includes employee and department metadata.

- **Additional data requests:**

Strict data governance enforcement via access policies. See access to database section below for more information.

- **Who will own/manage data**

Human resource.

- **Who will have access to database**

Only management and human resources personnel is allowed to access employee salary data as well as have read and write access. Everyone else would have read only access (except for salary data).

Data Architect Business Requirement

- **Estimated size of database**

The database will have 11 tables, it's initial record count is 205. Within next five years, with estimated growth of 20%, we will not require server resource / hardware upgrade. The record count within next five years is estimated to be approximately 364.

- **Estimated annual growth**

Expected yearly growth is 20%.

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

Salary data is restricted to be accessed only by top management and human resources personnel.

Data Architect Technical Requirement

- **Justification for the new database**

Scalability - The current data store does not support rapid growth of the data. **Data governance** - The way the data is currently stored, there is no way to enforce any restrictions as to who can access and/or edit the data.

- **Database objects**

Stage - intended for migration purposes only (stage table).

Staff - contains information about a person part of staff

Employment - employee and role/position description entities.

Department - holds department data as well as it's manager's ID

Department Manager - a mapping table that establishes relationship between department and staff (manager)

Staff Education - holds staff education level metadata.

Salary - restricted entity, accessible only by top management and human resources personnel.

Position - holds employment position's metadata.

Location, Address, City, State - location related dimension entities.

- **Data ingestion**

Data will be ingested with ETL approach. First the data will be loaded in landing (stage) table and then via SQL queries moved into the snowflake schema, that will be used for production once migration is complete.

Data Architect Technical Requirement

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

Ownership: Human resources department

User Access: Every employee will have read-only access (except salary entity). Human resources as well as top management will have read and write access.

- **Scalability**

Replication may be required as organisation grows, but is not an immediate concern. *Sharding will not be used due to its eventual consistency properties.*

- **Flexibility**

No additional measures are required.

- **Storage & retention**

Storage (disk or in-memory): 1 GB of space for data storage partition is sufficient.

Retention: Data needs to be kept for at least 7 years due to legal considerations.

- **Backup**

Daily incremental and weekly full backups.



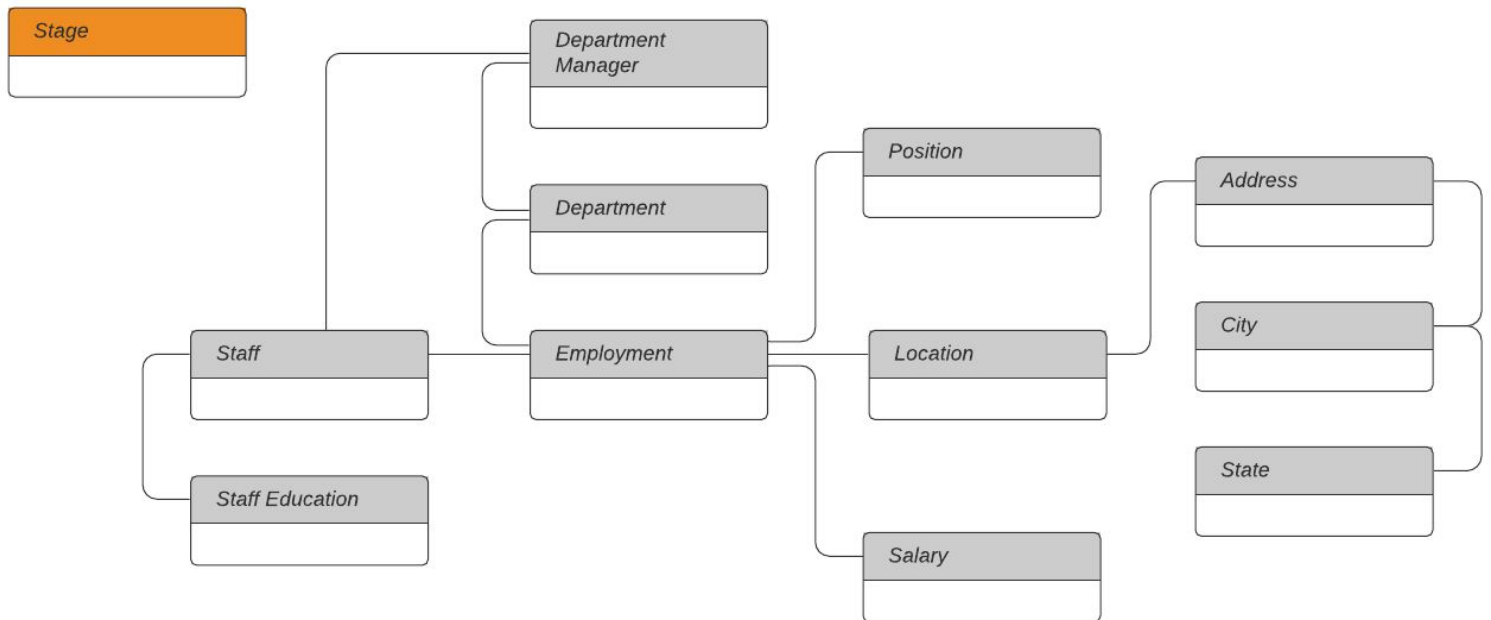
Step 2

Relational Database Design

Conceptual ERD

Project 1 - Conceptual ERD

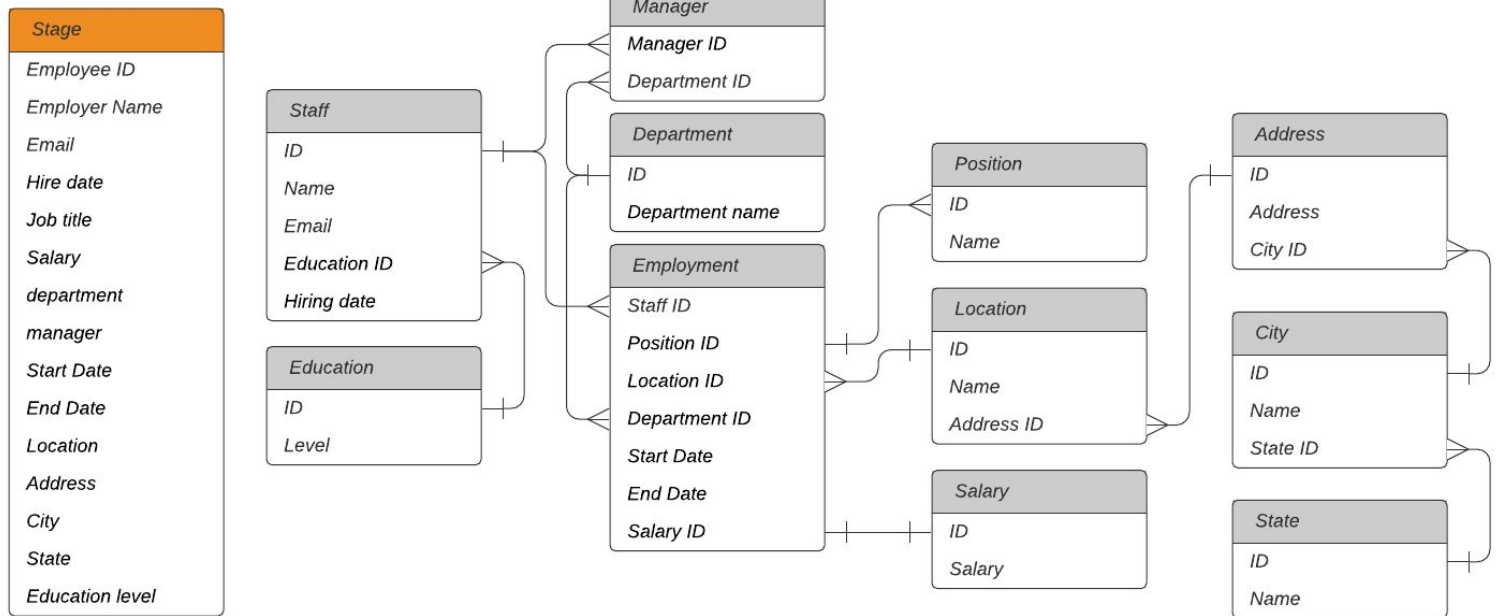
Jānis Pūris | June 28, 2021



Logical ERD

Project 1 - Logical ERD

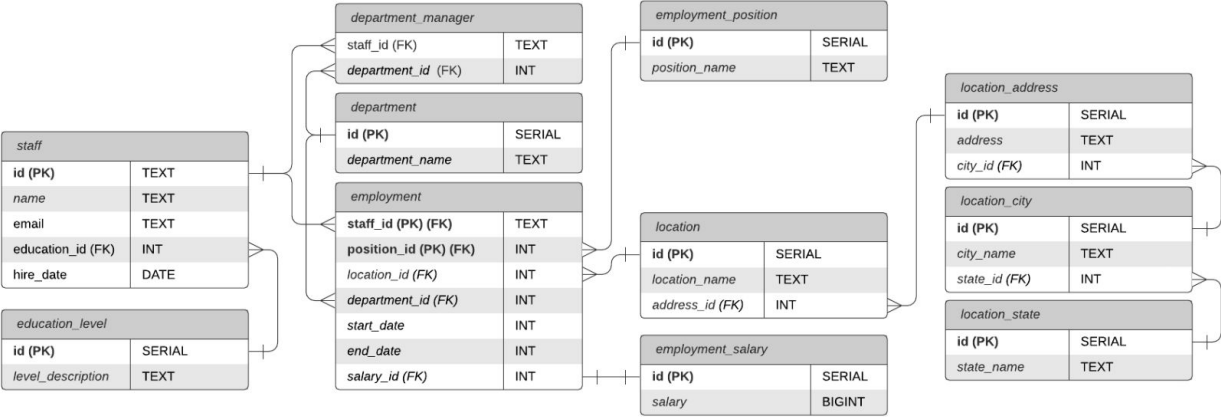
Jānis Pūris | June 28, 2021



Physical ERD

Project 1 - Physical ERD
Jānis Pūris | June 24, 2021

landing_stage	
emp_id	INT
emp_name	TEXT
email	TEXT
hite_date	DATE
job_title	TEXT
salary	TEXT
department	TEXT
manager	TEXT
start_date	DATE
end_date	DATE
location	TEXT
address	TEXT
city	TEXT
state	TEXT
education_level	TEXT





Step 3

Create A Physical
Database

CRUD

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

```
postgres=# \pset pager 0
Pager usage is off.
postgres=# SET search_path = human_resources;
SET
postgres=# SELECT
  s.id,
  s.name,
  ep.position_name,
  d.department_name
FROM employment e
JOIN employment_position ep ON e.position_id = ep.id
JOIN department d ON e.department_id = d.id
JOIN staff s ON e.staff_id = s.id;
```

id	name	position_name	department_name
E53895	Kumar Durairaj	Shipping and Receiving	Distribution
E15292	Melinda Fisher	Software Engineer	IT
E34816	Roseann Martineeti	Sales Rep	Product Development
E81369	Kevin Soltis	Network Engineer	IT
E16995	Wilson Martinez	Sales Rep	Product Development
E33486	Stephen Collucci	Legal Counsel	HQ
E60929	Tanya Maheshwari	Legal Counsel	Product Development
E47655	Cody Holland	Legal Counsel	IT
E87822	Anil Padala	Software Engineer	Product Development
E93734	Melissa DeMaio	Sales Rep	Product Development
E31931	Leo Manhanga	Shipping and Receiving	Distribution
E56444	Curtis Steward	Sales Rep	Sales
E39652	Kyle Guilmartin	Sales Rep	Product Development
E82137	Arup Das	Software Engineer	IT
E95199	Carlos Fernandes	Sales Rep	Product Development
E47926	Janice Canterbury	Network Engineer	Product Development
E20848	Edward Esler	Shipping and Receiving	Distribution
E10407	Darshan Rathod	Sales Rep	Product Development
E32359	Jen Frangias	Sales Rep	Sales
E20848	Edward Esler	Software Engineer	IT
E48637	Joey Fulkerson	Design Engineer	Product Development
E49459	Courtney Newman	Shipping and Receiving	Distribution
E10033	Jermaine Massey	Software Engineer	Product Development
E37389	Becky Weaver	Sales Rep	Sales
E96856	Tom Wilson	Legal Counsel	HQ
E67793	Fausto Recalde	Network Engineer	IT
E38997	Roseann Clemente	Sales Rep	Sales
E91075	Michah Vass	Network Engineer	IT
E98891	Doris Venama	Database Administrator	IT
E75344	Michael Kapper	Software Engineer	IT
E76053	Darryl Reamer	Legal Counsel	Product Development
E34496	Tony Hughes	Administrative Assistant	Sales
E62527	Barry Walsh	Network Engineer	IT
E67190	Nathan Hile	Sales Rep	Product Development
E22785	Tami Smith	Sales Rep	Sales
E56459	Raven Landis	Administrative Assistant	HQ
E87230	Sara Erwin	Sales Rep	Sales
E55855	Parker Williams	Sales Rep	Sales
E35053	Ashley Bergman	Administrative Assistant	Distribution
E35075	John Certa	Administrative Assistant	HQ
E94387	Erica Davis	Sales Rep	Product Development
E95214	Faheem Ahmed	Legal Counsel	IT
E77884	Conner Kinch	Manager	Product Development
E54196	Phil Wisneski	Sales Rep	Product Development
E87219	Jorge Moscoso	Administrative Assistant	Sales
E18697	Anita Deluise	Administrative Assistant	HQ
E79464	Anu Patel	Legal Counsel	HQ
E36988	Michelle Zietz	Shipping and Receiving	Distribution
E37246	Paulius Mikalainis	Database Administrator	IT
E26322	Raisa Paulson	Software Engineer	Product Development
E29652	Jennifer Westin	Software Engineer	Product Development
E84122	Congkhanh Nguyen	Software Engineer	IT
E93871	Travis Black	Sales Rep	Product Development
E63041	Allison Gentle	Manager	Distribution
E69297	Nilden Tutarar	Shipping and Receiving	Distribution
E93715	Charles Barker	Sales Rep	Sales
E99949	William Graf	Administrative Assistant	IT
E68807	Wes Tappan	Sales Rep	Sales
E41712	Elaine Podwika	Design Engineer	Product Development
E22197	Oliver Jia	Network Engineer	IT
E52489	Kelly Price	Shipping and Receiving	Distribution
E76443	Thunaja Polani	Database Administrator	IT
E65052	Leobrian Mason	Sales Rep	Sales
E96966	Lu Huang	Software Engineer	IT
E49025	Mark Flore	Database Administrator	IT
E51619	Tom Meola	Design Engineer	IT
E53406	Janice Mayzlik	Legal Counsel	HQ
E48884	Stacey Lewis	Network Engineer	IT
E16346	Jill Fram	Administrative Assistant	Product Development
E55880	Alex Warring	Shipping and Receiving	Distribution
E91182	Nick Gowen	Sales Rep	Product Development

CRUD

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

```
postgres=# INSERT INTO employment_position (position_name)
VALUES
    ('Web Programmer');
```

```
SELECT *
FROM employment_position
WHERE position_name ILIKE 'web%';
```

```
INSERT 0 1
```

```
id | position_name
```

```
----+-----
```

```
12 | Web Programmer
```

```
(1 row)
```

CRUD

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

```
postgres=# UPDATE employment_position
SET
    position_name = 'Web Developer'
WHERE position_name = 'Web Programmer';
UPDATE 1
postgres=# SELECT *
FROM employment_position
WHERE position_name ILIKE 'web%';
 id | position_name
----+-----
 12 | Web Developer
(1 row)
```


CRUD

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

```
postgres=# DELETE
FROM employment_position
WHERE position_name = 'Web Developer';
DELETE 1
postgres=# SELECT *
FROM employment_position
WHERE position_name ILIKE 'web%';
 id | position_name
----+-----
(0 rows)
```

CRUD

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

```
postgres=# SELECT
    d.department_name,
    COUNT(DISTINCT staff_id) AS employee_count
FROM employment e
    JOIN department d ON e.department_id = d.id
GROUP BY d.id;
```

department_name	employee_count
Product Development	70
HQ	13
Distribution	27
Sales	41
IT	52

(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.**

```
postgres=# SELECT
  s.name AS employee_name,
  ep.position_name AS job_title,
  d.department_name AS department,
  m.name AS manager_name,
  e.start_date AS position_start_date,
  e.end_date AS position_end_date
FROM employment e
JOIN staff s ON e.staff_id = s.id
JOIN employment_position ep ON e.position_id = ep.id
JOIN department d ON e.department_id = d.id
LEFT JOIN staff m ON e.manager_id = m.id
WHERE s.name = 'Toni Lembeck';
 employee_name | job_title           | department | manager_name | position_start_date | position_end_date
-----+-----+-----+-----+-----+-----
 Toni Lembeck   | Database Administrator | IT         | Jacob Lauber | 2001-07-18          |
 Toni Lembeck   | Network Engineer     | IT         | Jacob Lauber | 1995-03-12          | 2001-07-17
(2 rows)
```

CRUD

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

Create different roles and segment users into those roles.

One role would have read and write access (NO DDL's) to all tables in human_resources schema, while the other would have read only access to the schema's tables, except with no access to salary entity.



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

```
postgres=# CREATE OR REPLACE VIEW all_employee_data AS
SELECT
  e.staff_id AS "EMP_ID",
  s.name AS "NAME",
  s.email AS "EMAIL",
  s.hire_date AS "HIRE_DT",
  ep.position_name AS "JOB_TITLE",
  es.salary AS "SALARY",
  d.department_name AS "DEPARTMENT",
  m.name AS "MANAGER",
  e.start_date AS "START_DT",
  e.end_date AS "END_DT",
  l.location_name AS "LOCATION",
  la.address AS "ADDRESS",
  lc.city_name AS "CITY",
  ls.state_name AS "STATE",
  el.level_description AS "EDUCATION_LEVEL"
FROM employment e
JOIN staff s ON e.staff_id = s.id
LEFT JOIN staff m ON e.manager_id = m.id
JOIN employment_position ep ON e.position_id = ep.id
JOIN employment_salary es ON e.salary_id = es.id
JOIN department d ON e.department_id = d.id
JOIN location l ON e.location_id = l.id
JOIN location_address la ON l.address_id = la.id
JOIN location_city lc ON l.city_id = lc.id
JOIN location_state ls ON lc.state_id = ls.id
JOIN education_level el ON s.education_id = el.id;
```

```
graph LR
    Manager --> Employee
    Position --> Employee
    Department --> Employee
    Location --> Employee
    Address --> Location
    City --> Location
    State --> Location
    EducationLevel[Education Level] --> Employee
```

```
-- Information level of an education_id = el_id
SELECT * FROM all_employee_data;

-- Challenge 2: Create a stored procedure to generate first names, last names, and email addresses for all employees.
-- Create function to generate employee history:
CREATE FUNCTION employment_history()
RETURNS TABLE
AS
$$
BEGIN
  RETURN QUERY
  SELECT
    emp.emp_name,
    emp.hire_date,
    emp.position_name,
    emp.department_name,
    emp.manager_name,
    emp.start_date,
    emp.end_date
  FROM employment e
  JOIN staff s ON e.staff_id = s.id
  JOIN position p ON e.position_id = p.id
  JOIN department d ON e.department_id = d.id
  JOIN manager m ON e.manager_id = m.id
  JOIN location l ON e.location_id = l.id
  JOIN location_address la ON l.address_id = la.id
  JOIN location_city lc ON l.city_id = lc.id
  JOIN location_state ls ON lc.state_id = ls.id
  JOIN education_level el ON s.education_id = el.id;
END;
```

EMP_ID	NAME	EMAIL	HIRE_DT	JOB_TITLE	SALARY	DEPARTMENT	MANAGER	START_DT	END_DT	LOCATION	ADDRESS	CITY	STATE	EDUCATION_LEVEL
E53895	Kumar Duraiaj	kumar.durairaj@techcorp.com	2014-10-27	Shipping and Receiving	28788	Distribution	Allison Gentle	2014-10-27		West Coast	785 James Way	San Francisco	CA	No College
E15292	Melinda Fisher	melinda.fisher@techcorp.com	2011-02-06	Software Engineer	117958	IT	Jacob Lauber	2011-02-06		HQ	1 Tech ABC Corp Way	Dallas	TX	Bachelors Degree
E34816	Roseann Martineet	roseann.martineet@techcorp.com	2000-08-01	Sales Rep	82192	Product Development	Conner Kinch	2000-08-01		East Coast	165 Broadway	New York City	NY	Bachelors Degree
E01369	Kevin Soltis	kevin.soltis@techcorp.com	2019-05-29	Network Engineer	93362	IT	Jacob Lauber	2019-05-29		East Coast	165 Broadway	New York City	NY	Bachelors Degree
E16995	Wilson Martinez	wilson.martinez@techcorp.com	2001-02-28	Sales Rep	130161	Product Development	Conner Kinch	2001-02-28		HQ	1 Tech ABC Corp Way	Dallas	TX	Bachelors Degree
E33486	Stephen Colluci	stephen.colluci@techcorp.com	2000-08-02	Legal Counsel	153296	HQ	Tyrone Hutchison	2000-08-02		HQ	1 Tech ABC Corp Way	Dallas	TX	Doctorate
E06929	Tanya Maheshwari	tanya.maheshwari@techcorp.com	2013-07-03	Legal Counsel	195592	Product Development	Conner Kinch	2013-07-03		Midwest	1300 Nicollet Mall	Minneapolis	MN	Masters Degree
E47655	Cody Holland	cody.holland@techcorp.com	1997-11-27	Legal Counsel	207651	IT	Jacob Lauber	1997-11-27		East Coast	165 Broadway	New York City	NY	Doctorate
E87822	Anil Padala	anil.padala@techcorp.com	2014-04-02	Software Engineer	90884	Product Development	Conner Kinch	2014-04-02		East Coast	165 Broadway	New York City	NY	Bachelors Degree
E93734	Melissa DeMaio	melissa.demaio@techcorp.com	2005-06-21	Sales Rep	111114	Product Development	Conner Kinch	2005-06-21		East Coast	165 Broadway	New York City	NY	Bachelors Degree
E31931	Leo Manhanga	leo.manhanga@techcorp.com	2013-09-26	Shipping and Receiving	30286	Distribution	Allison Gentle	2013-09-26		Midwest	1300 Nicollet Mall	Minneapolis	MN	Some College
E56444	Curtis Steward	curtis.steward@techcorp.com	1999-06-10	Sales Rep	87817	Sales	Jennifer De La Garza	1999-06-10		East Coast	165 Broadway	New York City	NY	Masters Degree
E39652	Kyle Gullmartin	kyle.gullmartin@techcorp.com	2012-08-11	Sales Rep	71867	Product Development	Conner Kinch	2012-08-11		East Coast	165 Broadway	New York City	NY	Bachelors Degree
E02137	Arup Das	arup.das@techcorp.com	2006-10-04	Software Engineer	97840	IT	Jacob Lauber	2006-10-04		East Coast	165 Broadway	New York City	NY	Masters of Business Administration
E95199	Carlos Fernandes	carlos.fernandes@techcorp.com	2017-01-09	Sales Rep	153889	Product Development	Conner Kinch	2017-01-09		HQ	1 Tech ABC Corp Way	Dallas	TX	Masters Degree
E47926	Janice Canterbury	janice.canterbury@techcorp.com	2008-05-31	Network Engineer	85888	Product Development	Conner Kinch	2008-05-31		East Coast	165 Broadway	New York City	NY	Bachelors 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.

```
postgres=# DROP FUNCTION IF EXISTS employment_history;
CREATE FUNCTION employment_history(staff_name TEXT)
RETURNS TABLE
(
    employee_name TEXT,
    job_title TEXT,
    department TEXT,
    manager_name TEXT,
    position_start_date DATE,
    position_end_date DATE
)
AS $$
BEGIN
    RETURN QUERY
    SELECT
        s.name,
        ep.position_name,
        d.department_name,
        m.name,
        e.start_date,
        e.end_date
    FROM employment e
    JOIN staff s ON e.staff_id = s.id
    JOIN employment_position ep ON e.position_id = ep.id
    JOIN department d ON e.department_id = d.id
    LEFT JOIN staff m ON e.manager_id = m.id
    WHERE s.name = staff_name;
END;
$$
LANGUAGE plpgsql;
DROP FUNCTION
CREATE FUNCTION
postgres=# SELECT * FROM employment_history('Toni Lembeck');
```

employee_name	job_title	department	manager_name	position_start_date	position_end_date
Toni Lembeck	Database Administrator	IT	Jacob Lauber	2001-07-18	
Toni Lembeck	Network Engineer	IT	Jacob Lauber	1995-03-12	2001-07-17

(2 rows)

Standout Suggestion 3

Implement user security on the restricted salary attribute.

```
postgres=# CREATE ROLE management_role;
GRANT USAGE ON SCHEMA human_resources TO management_role;
GRANT USAGE, SELECT ON ALL SEQUENCES IN SCHEMA human_resources TO management_role;
GRANT SELECT, UPDATE, INSERT, DELETE ON ALL TABLES IN SCHEMA human_resources TO management_role;

CREATE ROLE employee_role;
GRANT USAGE ON SCHEMA human_resources TO employee_role;
GRANT SELECT ON ALL TABLES IN SCHEMA human_resources TO employee_role;
REVOKE SELECT ON TABLE employment_salary FROM employee_role;

CREATE USER employee_user WITH PASSWORD 'employee_user';
CREATE USER management_user WITH PASSWORD 'management_user';

GRANT management_role TO management_user;
GRANT employee_role TO employee_user;
```



Appendix

Additional Info

Please see following files on github repository

https://github.com/jpuris/udacity-da-nand-projects/tree/main/project_1

Step	Filename
DDL	1_create_schema.sql
ETL (staging)	2_load_stage.sql
ETL (load schema)	3_sql_etl.sql
CRUD	4_crud.sql
Challenges	5_challenges.sql