Project Title: EDB Software Company Database System

Name: Edgar Catalan

Student ID: 920829907

GitHub username: edgarjoel18

Milestone/Version	Date
M2V1	4/23/2021
M1V1	3/9/2021

Table of Contents

Section I: Project Description

There is a start up software company called XCompany and they just hired Bill as a software engineer to work on their various projects for their backend and frontend teams. Bill would be working with more than one software engineer, each who are part of a different team in Engineering department. Since Bill is part of various projects he will also work with various teams at XCompany from different departments like the UI/UX, business, and marketing teams. At XCompany every project may have multiple departments working together. However every employee is part of one department such as the engineering, business, or marketing team, but a project at XCompany can be made up of various teams and departments. The EDB database aims to keep track of every employees' personal information, their department, their teams in that department, their payroll information, their seniority level in the company, the Company perks they request, and their bonuses throughout the season. Also EDM allows every employee to have certain admin privileges based on their seniority level with the company and based on this level. Certain employees can give access to another employee to update their information update their information on EDB. For example, Bill has a meeting with the HR department to talk about dependents. Bill learns he is limited to at most 1 dependent. If Bill wants to change this property he will need an HR employee to give permission to edit this in EDB.

Section II: Use Cases

1) Since Bill is a recent hire to XCompany as a prerequisite to having his information in EDB. Bill has to provide his bank account, ssn, date of birth, phone number, emergency contacts, and at most 1 dependent. This can be done on an employee form on XCompany's employee app. Once Bill has entered this **information** the database generates an employee_id for authentication and Bill is officially an employee at XCompany. When Bill needs to update any of his personal information he can **log in to the Company's employee app with the unique employee_id and password and change his password**. **However, to change certain properties like dependents he needs to get authorization from an HR employee.**(functional requirement)
2) Bob is Bill's manager and he is able to enter Bill's roles, date hired, and seniority level at XCompany. Bob i**s able to access this information(Team management entity for managers)** on XCompany's **employee app using his employee_id and password**. Once he has logged in, **Bob has manager privileges and is only able to access Bill's responsibilities in the company. Whether he is working on a the backend, frontend, or full-stack group and what project he is working on. Bob is also able to assign team leads for the projects assigned to their department**.The app will send this information to EDB to the employee table and project table.
3) Betty is a Project Manager she is able t**o manipulate EDB through the employee app by logging in with her employee_id and password which authorizations her PM privileges**. **Once logged in she can choose the from different departments to work on a certain project(Project Management View)**. The employees assigned to a certain project are set by the manager in each department. Once Betty see's there is a team available she

can assign various teams to a project. Once Betty clicks the save button the information about a project can be save to a project table in EDB.

4) Becky works in HR and she has access to the employee app by logging in with her employee_id and password. Once logged in she is authorized as an HR employee and has access to Bill's personal information. Becky is able to give permission for Bill to update certain information like the number of dependents and other properties.
5) Christy is an Employee and has access to the company's perks in the office by entering their employee id and password through the employee application. Once entered the employee can request lunch and an uber to meet a business need or emergencies.
6) Emma is CEO and has the highest privileges to overlook and adjust EDB at any moment without requesting an authorization.
7) Trisha is a intern at XCompany and she able to use all the perks in the company and work in various teams. However since she is a intern she doesn't get full Health Insurance.
8) HurtEmployee is an Insurance company that provides Health Insurance to XCompany's Employees. They have been providing health insurance of all types to XCompany since becoming a medium sized startup company. They provide Health, Dental, and Vision Insurance to XCompany.

Section III: Database Requirements(Business Rules)

1) Employee

1.1. An Employee shall belong to only one Department

1.2. An Employee shall work on many Projects

1.3. An Employee shall work with many Teams

1.4. An Employee shall receive admin privileges based on seniority level

1.5. An Employee shall be able to log into the system any time with employee_id and password from many Devices

1.6. An Employee can request Company perks

1.7. An Employee has one Seniority Level

1.8. An Employee has many Employee Permissions on EDB Account

1.9. An Employee can have 1 Dependent under his account

2. Company

2.1. A Company shall have many Employees

2.2. A Company shall have many Departments

2.3. A Company shall have many Projects

2.4. A company shall have many Perks

2.5. A company shall have many Teams

2.6. A Company has many Seniority Levels

3. Department

3.1. A Department shall have many employees

3.2. A Department shall have many projects

3.3 A Department shall have many teams

3.4. A Department is a HR Department

4. Projects

4.1. A Project shall have be worked on by many departments

4.2. A Project shall be managed by one to many Team

4.3 A Project shall be worked on by many Employees

4.4 A Project shall be worked by many Teams

5. Team

4.1. A Team shall belong to one Department

4.2. A Team shall have many Employees

4.3. A Team belongs to one Company

4.3 A team has a unique id

4.4 A Team gets many bonuses

6. Perks

6.1. A perk shall be requested by many Teams when being logged into employee app

6.2. A perk belongs to one Company

6.3. A perk is provided by an Doordash or Uber

7. Permission

7.1. A Permission belongs to many Employee

7.2. A Permission has only one Seniority Level

8. Seniority Level

8.1 Seniority level belongs to many Company

8.2 Seniority level belongs to one Employee

8.3. All Seniority Level receive many Bonuses

8.4 Seniority level has unique_id

8.5 Seniority level references employee_id

10. Payroll

10.1. Payroll can be Direct Deposit

10.2 Payroll can be Checks

10.3 Payroll Managed by many HR Employees

11. Bonuses

11.1 Bonuses are given to all Seniority Level

12. Insurance

12.1 Insurance belongs to many Companies

12.2 Insurance belong to Employees that are not interns

12.3 Insurance is Vision, Dental, and Health Insurance

13. EDB Account

13.1 EDB Account created by many Employees

13.2 EDB Account has many Authorization

13.3 EDB Account is Logged in by many devices

13.4 EDB Account is a Manager Account

13.5 EDB Account is a Intern Account

13.6 EDB Account is a HR Account

13.7 EDB Account is a Base Employee Account

13.8 EDB Account is a Team Lead Account

14. Devices:

14.1. A Device is used by many Employees

14.2. A Device logs into one EDB Account

15. HR Department:

15.1. HR Department manages one Payroll

15.2. HR Department is a Department

Section IV: Detailed List of Main Entities, Attributes, and Keys

9) Company(Strong)

- Company_Id: key, numeric
- Name: composite, alphanumeric
- Company_Type: alphanumeric
- Date_First_Using_EDB: Date

2. Department(Strong)

- Department_Id: key, numeric
- Project_Id: weakKey, numeric
- Team_Id: weakKey, numeric
- Name: alphanumeric

3. Team(Strong)

- Team_Id: key, numeric
- Employee_Id: weakKey, numeric
- Department: weakKey, numeric
- Project_Id: weakKey, numeric
- Name: alphanumeric

4. Seniority Level(Strong):

- Seniority_Id: key, numeric
- Employee_Id: weakKey, numeric
- Level: alphanumeric

5. Permission(Strong):

- Permission_Id: key, numeric
- Employee_Id: weakKey, numeric
- Name: alphanumeric

6. Insurance Company(Strong):
   - Insurance_Id: key, numeric
   - Employee_Id: weakKey, numeric
   - Dept_Id: weakKey, numeric
   - Name: alphanumic
7. Perks(Strong):
   - Perks_Id: key, numeric
   - Team_Id: weakKey, numeric
   - Uber: Boolean
   - Doordash: Boolean
8. Payroll(Strong):
   - Payroll_Id: key, numeric
   - Employee_Id: weakKey, numeric
   - Deptartment_Id: weakKey, numeric
   - Team_Id: weakKey, numeric
   - Amount: numeric
9. Devices(Strong):
   - MAC: key, alphanumeric
10. Employee(Weak):
    - Employee_Id: key, numeric

- Department_Id: weakKey, numeric
- Team_Id: weakKey, numeric
- Name: composite, alphanumeric
- Dependent: Boolean
- SSN: numeric
- Date_Of_Birth: numeric
- Address: composite, alphanumeric

11. Project(Weak):
    - Project_Id: key, numeric
    - Employee_Id: weakKey, numeric
    - Department_Id: weakKey, numeric
    - Team_Id: weakKey, numeric
12. EDB Account(Weak):
    - EDB_Account_Id: key, numeric
    - Employee_Id: weakKey, numeric
    - Department_Id: weakKey, numeric
    - Project_Id: weakKey, numeric
    - Manager Account: alphanumeric
    - Intern Account: alphanumeric
    - HR Account: alphanumeric
    - Base Employee Account: alphanumeric
    - Team Lead Account: alphanumeric
13. Authorization(Weak):

- Authorization_Id: key, numeric
- Permission_Id: weakKey, numeric
- Employee_Id: weakKey, numeric

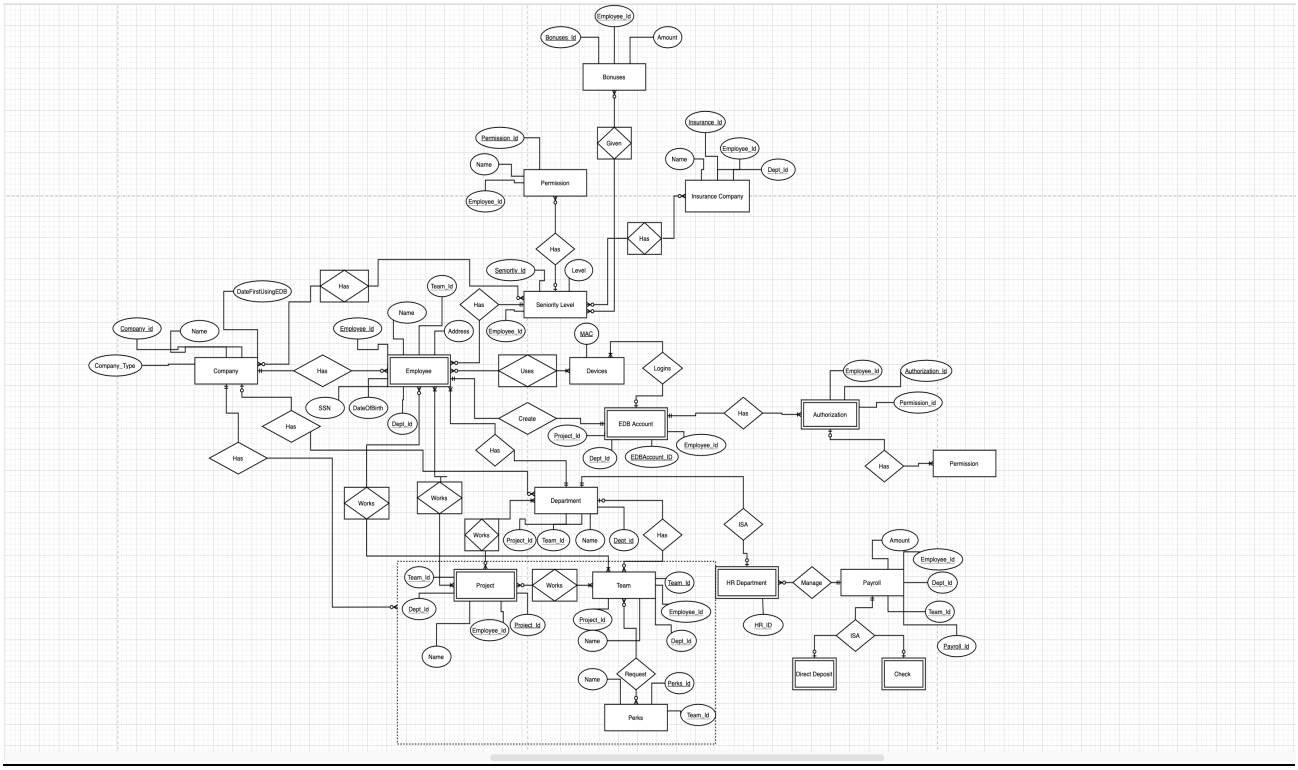
14. HR Department(Weak):

- HR_Department_Id: key, numeric

15. Bonuses(Strong):

- Bonuses_Id: key, numeric
- Employee_Id: weakKey, alphanumeric
- Amount: numeric

Section V: Entity Relationship Diagram(ERD)



Section VI: Testing Table

Rule	Entity A	Relation	Entity B	Cardinality	Pass/Fail	Error Description	
1	EDB Account	ISA	Manager Account, Intern Account, etc…	1- to - 1	Fail	The tables for the different types of employees could be a waste of space because I was making a table for each different type of employee. An easy fix can be adding these as columns to the EDB Account	
2	Authorization	Has	Permissions	1 - to - M	Fail	My authorization table had no relationship to access the properties of the permissions table. Therefore, I added the relationship between the two tables also I did not have permission_Id as a foreign key which is also necessary to have	
3	Perks	ISA	Doordash,Uber	1 - to - 1	Fail	Having the extra tables for Perks could be a waste of memory when it could become just columns.	
5	Company	Has	Employee	1 - to - M	Pass	None	
6	Company	Has	Departments	1 - to - M	Pass	None	
7	Company	Has	Projects	1 - to - M	Pass	None	
8	Company	Has	Perks	1 - to - M	Pass	None	
9	Company	Has	Teams	1 - to - M	Pass	None	
10	Employee	Has	Department	1 - to - 1	Pass	None	
11	Employee	Works	Projects	M - to - M	Pass	None	
12	Employee	Works	Teams	M - to - M	Pass	None	
14	Employee	Has	Company	M - to - 1	Pass	None	
15	Employee	Has	Seniority Level	M - to - 1	Pass	None	

16	Department	Has	Employees	1 - to - M	Pass		
17	Department	Has	Projects	M - to - M	Pass		
18	Department	Has	Teams	1 - to - M	Pass		
20	Project	Has	Department	M - to - M	Pass		
21	Project	Works	Team	M - to - M	Pass		
22	Project	Has	Employees	M - to - M	Pass		
23	Team	Belongs	Department	M - to - 1	Pass		
24	Team	Has	Employees	M - to - M	Pass		
25	Team	Belongs	Company	M - to - 1	Pass		
26	Team	Given	Bonuses	M - to - M	Pass		
27	Perks	Requested	Teams	M - to - M	Pass		
28	Perks	Belongs	Company	M - to - 1	Pass		
29	Perks	ISA	Doordash,Uber	1 - to - 1	Fail	This is another example of unnecessary usage of memory. We could turn these two tables into columns	
30	Permission	Has	Seniority Level	M - to - 1	Pass		
31	Seniority Level	Belongs	Company	M - to - M	Pass		
32	Seniority Level	Has	Employee	1 - to - M	Pass		
33	Seniority Level	Given	Bonuses	M - to - M	Pass		
34	Payroll	ISA	Direct Deposit, Check	1 - to - 1	Pass		
35	Payroll	Managed	HR Employee	1 - to - M	Pass		
36	Bonuses	Given	Seniority Level	M - to - M	Pass		
37	Insurance	Belongs	Company	M - to - M	Pass		
38	Insurance	Belongs	Seniority Level	M - to - M	Pass		
39	Insurance	ISA	Vision,Dental, Health Insurance	1 - to - 1	Fail	These extra tables can save memory by becoming	
40	EDB Account	Created	Employee	1 - to - M	Pass		

41	EDB Account	Logins	Device	1 - to - M	Pass		
42	EDB Account	Has	Authentication	1 - to - M	Pass		

Section VII: Database Model/EER:

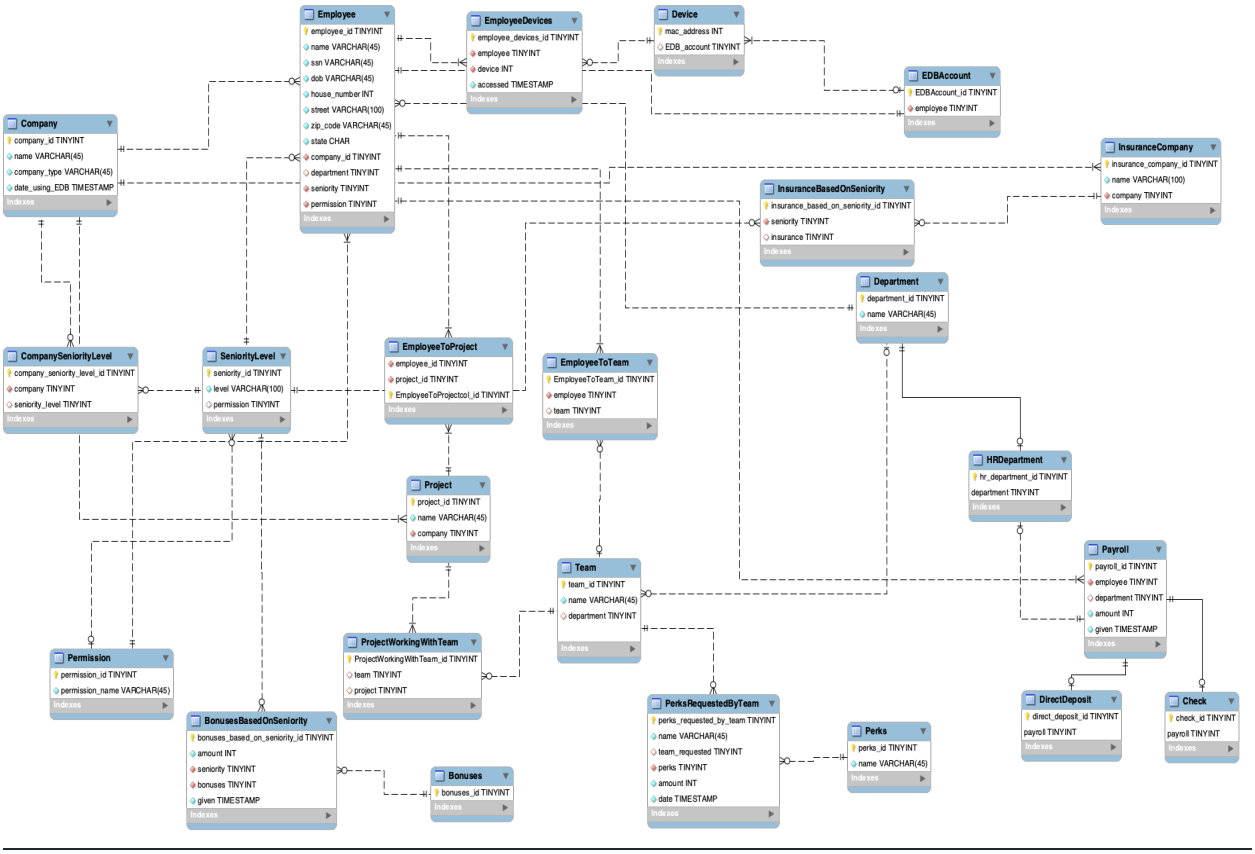


Table	FK	On Delete	On Update	Comment
Employee	Company	Cascade	Cascade	If a company gets deleted then the employees from that company get deleted

Employee	Department	Set Null	Cascade	If a department gets deleted then the Employee's department gets set to null until it is updated again
Employee	Seniority	Cascade	Cascade	If a Seniority is removed from the Employee the Employee is no longer working for the company
Employee	Permission	Cascade	Set null	If an Employee does not have a permission id then it is null until they make a EDB account
EmployeeDevices	Employee	Cascade	Cascade	If an Employee gets removed from the Employee devices then they don't have a device registered to an EDB account
EmployeeDevices	Devices	Set null	Cascade	If a device is removed from an employee. It is set as null until they add a new device or re-add the same device
Devices	EDB Account	Cascade	Cascade	If a EDB account is deleted then all the devices used to log into the EDB Account is deleted
EDB Account	Employee	Cascade	Cascade	If an Employee is removed from the Company

				then their account is removed as well
Project	Company	Cascade	Cascade	A project belongs to a company. Therefore, if a Company is removed then a project from that company is removed
EmployeeToProject	Employee	Set null	Cascade	If an Employee is removed from a project then project has a null for an employee
EmployeeToProject	Project	Cascade	Cascade	If a project is removed then all the employees working on that project get removed
Team	Department	Set null	Cascade	If a department is removed from that team. Then the team belongs to a null department until it is updated
EmployeeToTeam	Employee	Set null	Cascade	If an Employee is removed from a Team then that Team has a null for employee
EmployeeToTeam	Team	Cascade	Cascade	If a Team is deleted then the Employees from that team get removed as well
ProjectWorkingWithTeam	Project	Set null	Cascade	If a project gets deleted then the Team is working on null project

				until it is updated
ProjectWorkingWithTeam	Team	Cascade	Cascade	If a team gets deleted then the project gets deleted as well
DepartmentWorkingWithProject	Department	Cascade	Cascade	If a Department gets deleted then the project in the table gets deleted as well
DepartmentWorkingWithProject	Project	Set null	Cascade	If a Project gets removed then the Department is working on null projects
PerksRequestedByTeam	Team	Set null	Cascade	If a team is removed from the Perks then the Team who ordered the perk is set to null just to keep track of how many times perks are ordered throughout the company to
PerksRequestedByTeam	Perks	Cascade	Cascade	If a Perks is removed then the whole request is removed from the table
CompanyWithSeniorityLevel	Seniority	Set null	Cascade	If a Seniority Level is removed from the Company then the Company as null Seniority Levels but the Company still exisits
CompanyWithSeniorityLevel	Company	Cascade	Cascade	If a Company is removed then

				the Seniority Levels are removed as well
BonusesBasedOnSeniority	SeniorityLevel	Cascade	Cascade	If a Seniority is removed then there is no Bonuses to be given to the Employees of this seniority.
BonusesBasedOnSeniority	Bonuses	Cascade	Cascade	If there are no Bonuses then there is nothing to give the Employees with Certain Seniority
InsuranceCompany	Company	Cascade	Cascade	If there is no Company then the insurance Company belongs to no company
InsuranceBasedOnSeniority	Seniority Level	Cascade	Cascade	If Seniority is removed then there is no insurance.
InsuranceBasedOnSeniority	InsuranceCompany	Set null	Cascade	If insurance company is null then Seniority Level will be null for Insurance
Bonuses	Provided no FK it is in the BonusesBasedOnSeniority table	None	None	This table needs a name attribute but when a bonus is created it is assigned to a team in the relationship table
BonusesBasedOnSeniority	Seniority, Bonuses, Team(missing, I need to add)	Set Null, Set Null, Set Null	Cascade, Cascade, Cascade	Since Seniority and Bonuses have a many-to-many relationship.

				When a bonus is created a trigger can set all the information about the bonus given in the middle table
SeniorityLevel	Permission	Set Null	Cascade	When a seniority level is made it can have many Permissions and a seniority level is unique to a company
CompanySeniorityLevel	Company, Seniority_level	Set Null, SetNull	Cascade, Cascade	When a Seniority level is created a trigger is created to set to which Company the Seniority Level belongs to.
Department	Company(needs to be added)	Cascade	Cascade	When a Company gets deleted a department gets deleted as well.
HR Department	Department	Cascade	Cascade	If the Department gets deleted then the HR department gets deleted as well.
Payroll	Employee, HRDepartment	Cascade, Cascade	Cascade, Cascade	Payroll is managed by a Employee who works in Human Resources. If either of these tables gets deleted then there is no one to send a payroll to an Employee
Company	None	None	None	foreign keys reference

				Company rather than having multiple columns
Check	Payroll	Cascade	Cascade	If there is no payroll sent to employees there can't be any checks sent so if Payroll is deleted the Check table is deleted as well
Direct Deposit	Payroll	Cascade	Cascade	If Payroll is deleted then Direct Deposit can be sent to Employees

Section VIII: Testing:

Entity	SQLQuery	Pass/Fail	Error Description	Possible Solution
Company	Delete	Pass	None	None
Company	Update	Pass	None	None
Employee	Update	Fail	Error Code: 1406	I might need to change the datatype of the column
Employee	Delete	Fail	I was not even allowed to make an insert	Need to fix the column datatype for state
Department	Update	Pass	None	None
Department	Delete	Pass	None	None
Team	Delete	Pass	None	None
Team	Update	Pass	None	None

Project	Update	Pass	None	None
Project	Delete	Pass	None	None
SeniorityLevel	Update	Pass	None	None
SeniorityLevel	Delete	Fail	Deletes parent reference	Need to fix the weak keys
CompanyWithSeniorityLevel	Update	Pass	None	None
CompanyWithSeniorityLevel	Delete	Pass	None	None
BonusesBasedOnSeniority	Update	Pass	None	None
BonusesBasedOnSeniority	Delete	Pass	None	None
SeniorityLevel	Update	Pass	None	None
SeniorityLevel	Delete	Pass	None	None
CompanySeniorityLevel	Update	Pass	None	None
CompanySeniorityLevel	Delete	Pass	None	None
Department	Update	Pass	None	None

Department	Delete	Pass	None	None
HR Department	Update	Pass	None	None
HR Department	Delete	Pass	None	None
Payroll	Update	Pass	None	None
Payroll	Delete	Pass	None	None
Company	Update	Pass	None	None
Company	Delete	Pass	None	None
Check	Update	Fail	None	None
Check	Delete	Fail	Error in foreign key constraint of table /#sql-7fa_247a: (`my_foreing_key`) references `table` (`id`) on delete cascade:	Does not delete the reference to employee
Direct Deposit	Pass	None	None	None

Direct Deposit	Delete	Fail	Error in foreign key constraint of table /#sql-7fa_247a: (`my_foreing_key`) references `table` (`id`) on delete cascade:	I need to fix the foreign keys in this entities