

Updated Components from checkpoint 1 of (2023287):

Schema Design –

**Employee Table -**

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Constraints** |
| EmployeeID | INT | Primary Key, Auto-Increment |
| FirstName | Varchar(255) | Not Null |
| LastName | Varchar(255) | Not Null |
| DateOfBirth | Date | Not Null |
| Gender | Varchar(255) | Not Null |
| AddressID | INT | Foreign Key |
| HierarchyID | INT | Foreign Key, Auto Increment |
| ContactID | INT | Foreign Key |
| DepartmentID | INT | Foreign Key |
| SalaryID | INT | Foreign Key |
| EmploymentStatusID | INT | Foreign Key |
| StatusID | INT | Foreign Key |
| JobID | INT | Foreign Key |

**Address Table -**

|  |  |  |
| --- | --- | --- |
| Field | Type | Constraints |
| AddressID | INT | Primary Key, Auto Increment |
| Street | Varchar(255) | Not Null |
| City | Varchar(255) | Not Null |
| Province | Varchar(255) | Not Null |
| Eircode | Varchar(255) | Null |

**Jobs Table -**

|  |  |  |
| --- | --- | --- |
| Field | Type | Constraints |
| JobID | Integer | Primary Key, Auto Increment |
| Language | Varchar(100) | Not Null |
| RequiredSkills | Text | Not Null |
| QualificationLevel | Varchar(255) | Not Null |

**Departments Table -**

|  |  |  |
| --- | --- | --- |
| Field | Type | Constraints |
| DepartmentID | Integer | Primary Key, Auto-Increment |
| DepartmentName | Varchar(255) | Not Null |
| Location | Varchar(255) | Not Null |
| NumberOfEmployees | INT |  |

**Contact Table-**

|  |  |  |
| --- | --- | --- |
| Field | Type | Constraints |
| ContactID | INT | Primary Key, Auto Increment |
| Email | Varchar(255) | Unique, Not Null |
| Country | Varchar(255) | Not Null |
| PhoneNumber | Varchar(255) | Unique, Not Null |

**Managerial Hierarchy Table -**

|  |  |  |
| --- | --- | --- |
| Field | Type | Constraints |
| HierarchyID | INT | Primary Key, Auto Increment |
| ManagerID | INT | Not Null |
| SubordinateID | INT | Not Null |
| AssignmentDate | Date | Not Null |

**Salary Table -**

|  |  |  |
| --- | --- | --- |
| Field | Type | Constraints |
| SalaryID | INT | Primary Key, Auto Increment |
| EmployeeID | INT | Not Null |
| Amount | Decimal (10,2) | Not Null |
| EffectiveDate | Date | Not Null |
| SalaryType | Varchar(255) | Not Null |

**EmploymentStatus Table -**

|  |  |  |
| --- | --- | --- |
| Field | Type | Constraints |
| StatusID | INT | Primary Key, Auto Increment |
| EmployeeID | INT | Not Null |
| Status | Varchar(255) | Not Null |
| StartDate | Date | Not Null |
| EndDate | Date | Null |

**CHEN DIAGRAM -**

A diagram of a diagram

Description automatically generated

**Entities and Their Attributes:**

**Employee:**

**Attributes:** EmployeeID (Primary Key), FirstName, LastName, DateOfBirth, Gender, and other related personal details.

The Employee entity is central in the diagram and has relationships with several other entities to describe job, contact, managerial hierarchy, employment status, and salary details.

**Jobs:**

**Attributes:** JobID (Primary Key), Title, RequiredSkills, Language.

This entity captures information about the various job roles within the organization, including the skills required and languages spoken.

**Department:**

**Attributes:** DepartmentID (Primary Key), DepartmentName, ManagerID (Foreign Key referencing Employee).

Represents different departments within the organization. Each department is managed by an employee who acts as the manager.

**Address:**

**Attributes:** AddressID (Primary Key), Street, City, Province, Eircode.

Stores address information where each employee lives; thus, it connects to the Employee entity showing where each employee resides.

**Contact:**

**Attributes:** ContactID (Primary Key), Email, Phone Number, Country.

Manages contact details for each employee, allowing for the storage of multiple contact methods per employee.

**Managerial Hierarchy:**

**Attributes:** HierarchyID (Primary Key), ManagerID, SubordinateID, AssignmentDate.

Details the reporting structure within the organization, specifying who reports to whom and the assignment date of the reporting relationship.

**Salary:**

**Attributes:** SalaryID (Primary Key), EmployeeID (Foreign Key), Amount, EffectiveDate, SalaryType.

Keeps track of salary information for each employee, including the amount, type (e.g., annual, hourly), and the date from which the salary is effective.

**Employment Status:**

**Attributes:** StatusID (Primary Key), EmployeeID (Foreign Key), Status, StartDate, EndDate.

Records changes in employment status over time, such as active, on leave, etc., with corresponding start and end dates.

**First Normal Form (1NF) for CA2**

To achieve the First Normal Form (1NF), we ensure that all fields in each table contain only atomic values and there are no repeating groups. Here’s how your schema achieves 1NF based on the provided structure:

**Employee Table**

* **Fields**: EmployeeID, FirstName, LastName, DateOfBirth, Gender, AddressID, HierarchyID, ContactID, DepartmentID, SalaryID, EmploymentStatusID, StatusID, JobID
* **Normalization**: Each field is designed to hold a single value per field, per record, ensuring atomicity. All ID fields serve as foreign keys to link with other tables, which avoids embedding complex data within the table. The auto-increment nature of EmployeeID ensures that each record is unique.

**Address Table**

* **Fields**: AddressID, Street, City, State, Eircode
* **Normalization**: Each address attribute is atomic, with AddressID as a unique identifier that auto-increments.

**Jobs Table**

* **Fields**: JobID, Language, RequiredSkills, QualificationLevel
* **Normalization**: Each field is atomic. JobID is unique and auto-incrementing, which properly indexes job-related data.

**Departments Table**

* **Fields**: DepartmentID, DepartmentName, Location, NumberOfEmployees
* **Normalization**: All fields are atomic, with DepartmentID providing a unique, auto-incrementing key for each department.

**Contact Table**

* **Fields**: ContactID, Email, Country, PhoneNumber
* **Normalization**: This table ensures that each contact detail is stored once per record, with ContactID as a primary key and all other fields providing unique, atomic information about each contact instance.

**Managerial Hierarchy Table**

* **Fields**: HierarchyID, ManagerID, SubordinateID, AssignmentDate
* **Normalization**: Each field is atomic, with HierarchyID uniquely identifying each managerial relationship, ensuring no repeating groups.

**Salary Table**

* **Fields**: SalaryID, EmployeeID, Amount, EffectiveDate, SalaryType
* **Normalization**: All fields are atomic with SalaryID as a unique identifier, linking salary records uniquely to employees via EmployeeID.

**Employment Status Table**

* **Fields**: StatusID, EmployeeID, Status, StartDate, EndDate
* **Normalization**: Each field is atomic, with StatusID uniquely indexing each status record; EndDate is nullable to accommodate ongoing statuses.



Address Table (1NF) –

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| AddressID | Street | City | |  | | --- | |  |   Province | Eircode |
| AD001 | 22 Cherry Lane | Dublin | Leinster | D02 X123 |
| AD002 | 48 St. Patrick’s Street | Cork | Munster | C12 Y456 |
| AD003 | Abbey Road | Dublin | Leinster | D91 Z789 |
| AD004 | 15 Oak Circle | Limerick | Munster | V94 R234 |
| AD005 | 77 New Street | Nass | Leinster | N86 Z345 |
| AD006 | 65 High Road | Galway | Connacht | W91 A567 |

Jobs Table -

|  |  |  |  |
| --- | --- | --- | --- |
| JobID | Language | RequiredSkills | QualificationLevel |
| J001 | English | Programming & Software Development | Bachelors |
| J002 | English | Data Management & Analytics | Masters |
| J003 | English | System Administration & Networking | Masters |
| J004 | English | Web Development | Bachelors |
| J005 | English | Cybersecurity | Bachelors |
| J006 | English | UI/UX Design | Masters |

Department Table -

|  |  |  |  |
| --- | --- | --- | --- |
| DepartmentID | DepartmentName | Location | NumberOfEmployees |
| D001 | Information Technology (IT) | Dublin | 4 |
| D002 | Finance and Accounting | Dublin | 4 |
| D003 | Data Analytics and Management | Dublin | 4 |
| D004 | Web and Digital Design | Dublin | 4 |
| D005 | Cybersecurity | Dublin | 4 |
| D006 | Web and Digital Design | Dublin | 4 |

Contact Table -

|  |  |  |  |
| --- | --- | --- | --- |
| ContactID | Email | Country | PhoneNumber |
| C001 | Emma.clarkson90@gmail.com | Ireland | +353 83123 4567 |
| C002 | Noah.greg88@gmail.com | Ireland | +353 85 234 5678 |
| C003 | Olivia.steve92@gmail.com | Ireland | +353 83 345 6789 |
| C004 | Liam.hendy85@gmail.com | Ireland | +353 85 456 7890 |
| C005 | Ava.sinclair93@gmail.com | Ireland | +353 86 567 8901 |
| C006 | Ethan.murph86@gmail.com | Ireland | +353 87 678 9012 |

Managerial Hieracrchy Table –

|  |  |  |  |
| --- | --- | --- | --- |
| HierarchyID | ManagerID | SubordinateID | AssignmentDate |
| H001 | M001 | S001 | 2024-01-15 |
| H002 | M002 | S002 | 2024-01-15 |
| H003 | M003 | S003 | 2024-02-01 |
| H004 | M004 | S004 | 2024-02-01 |
| H005 | M005 | S005 | 2024-03-20 |
| H006 | M006 | S006 | 2024-03-20 |

Salary Table (1NF)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SalaryID | EmployeeID | Amount | EffectiveDate | SalaryType |
| SA01 | EM001 | 50000 | 2019-01-01 | Annual |
| SA02 | EM002 | 42000 | 2019-01-01 | Annual |
| SA03 | EM003 | 28 | 2019-01-01 | Hourly |
| SA04 | EM004 | 3000 | 2019-01-01 | Monthly |
| SA05 | EM005 | 60000 | 2019-01-01 | Annual |
| SA06 | EM006 | 35 | 2019-01-01 | Hourly |

Employment Status Table (1NF)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| StatusID | EmployeeID | Status | StartDate | EndDate |
| ST01 | EM001 | Active | 2019-11-21 | Null |
| ST02 | EM002 | On Leave | 2019-10-05 | Null |
| ST03 | EM003 | Active | 2019-11-07 | Null |
| ST04 | EM004 | On Leave | 2019-11-14 | Null |
| ST05 | EM005 | Active | 2019-11-21 | Null |
| ST06 | EM006 | Active | 2019-11-28 | Null |

**Understanding Second Normal Form (2NF)**

**2NF Criteria:**

The table is in 1NF (already achieved as explained in previous responses).

All non-key attributes must be fully dependent on the primary key. This means that every attribute must depend on the whole key for composite keys. However, since our tables have only single primary keys, this condition focuses on ensuring that all fields in a table are related directly to the primary key without any partial dependency.

**Detailed 2NF Analysis for Each Table**

**Employee Table**

Primary Key: EmployeeID

Fields: FirstName, LastName, DateOfBirth, Gender, AddressID, HierarchyID, ContactID, DepartmentID, SalaryID, EmploymentStatusID, StatusID, JobID

2NF Compliance: Each attribute in the Employee table is dependent on EmployeeID. There are no composite keys, and no partial dependencies exist. Every non-key attribute is fully functionally dependent on the primary key (EmployeeID).

**Address Table**

Primary Key: AddressID

Fields: Street, City, State, Eircode

2NF Compliance: All attributes are dependent solely on AddressID. Since AddressID is the only key and no attribute depends on a subset of the key (because there isn't one), this table meets 2NF.

**Jobs Table**

Primary Key: JobID

Fields: Language, RequiredSkills, QualificationLevel

2NF Compliance: Similar to the Address table, all attributes in the Jobs table depend only on the primary key JobID. There are no partial dependencies, thus aligning with 2NF requirements.

**Departments Table**

Primary Key: DepartmentID

Fields: DepartmentName, Location, NumberOfEmployees

2NF Compliance: Each attribute here depends solely on DepartmentID. This table meets the 2NF criteria as there are no attributes that are dependent on a subset of a primary key.

**Contact Table**

Primary Key: ContactID

Fields: Email, Country, PhoneNumber

2NF Compliance: All the fields are fully dependent on ContactID, ensuring there is no partial dependency, which is in line with 2NF.

Managerial Hierarchy Table

Primary Key: HierarchyID

Fields: ManagerID, SubordinateID, AssignmentDate

2NF Compliance: As with other tables, each field depends on the primary key HierarchyID, and no partial dependencies are present.

**Salary Table**

Primary Key: SalaryID

Fields: EmployeeID, Amount, EffectiveDate, SalaryType

2NF Compliance: All fields in the Salary table are dependent on the primary key SalaryID, with no partial dependencies. EmployeeID here serves as a foreign key linking to the Employee table, not as a part of a composite key.

**Employment Status Table**

Primary Key: StatusID

Fields: EmployeeID, Status, StartDate, EndDate

2NF Compliance: All fields depend directly on StatusID. EmployeeID is a foreign key, not part of a composite key, thus avoiding any partial dependency issues.

**Understanding Third Normal Form (3NF)**

**3NF Criteria:**

A table is in 1NF and 2NF.

No non-key attribute is dependent on another non-key attribute (eliminating transitive dependencies).

**Detailed 3NF Analysis for Each Table in Your Schema**

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Constraints** |
| EmployeeID | INT | Primary Key, Auto-Increment |
| FirstName | Varchar(255) | Not Null |
| LastName | Varchar(255) | Not Null |
| DateOfBirth | Date | Not Null |
| Gender | Varchar(255) | Not Null |
| AddressID | INT | Foreign Key |
| HierarchyID | INT | Foreign Key, Auto Increment |
| ContactID | INT | Foreign Key |
| DepartmentID | INT | Foreign Key |
| SalaryID | INT | Foreign Key |
| EmploymentStatusID | INT | Foreign Key |
| StatusID | INT | Foreign Key |
| JobID | INT | Foreign Key |

**3NF Review:** All attributes depend directly on EmployeeID. Attributes such as DepartmentID, JobID, and SalaryID are foreign keys linking to other tables, not attributes derived from another non-key attribute. Therefore, no transitive dependencies are apparent, meeting 3NF.

**2. Address Table**

|  |  |  |
| --- | --- | --- |
| Field | Type | Constraints |
| AddressID | INT | Primary Key, Auto Increment |
| Street | Varchar(255) | Not Null |
| City | Varchar(255) | Not Null |
| State | Varchar(255) | Not Null |
| Eircode | Varchar(255) | Null |

**3NF Review**: All fields are dependent on AddressID, and there are no fields in the table that depend on other non-key attributes. This table meets 3NF.

**3. Jobs Table**

|  |  |  |
| --- | --- | --- |
| Field | Type | Constraints |
| JobID | Integer | Primary Key, Auto Increment |
| Language | Varchar(100) | Not Null |
| RequiredSkills | Text | Not Null |
| QualificationLevel | Varchar(255) | Not Null |

**3NF Review:** Language, RequiredSkills, and QualificationLevel all depend directly on JobID. There is no dependence on non-key attributes, thus meeting 3NF.

**4. Departments Table**

|  |  |  |
| --- | --- | --- |
| Field | Type | Constraints |
| DepartmentID | Integer | Primary Key, Auto-Increment |
| DepartmentName | Varchar(255) | Not Null |
| Location | Varchar(255) | Not Null |
| NumberOfEmployees | INT |  |

**5. Contact Table**

|  |  |  |
| --- | --- | --- |
| Field | Type | Constraints |
| ContactID | INT | Primary Key, Auto Increment |
| Email | Varchar(255) | Unique, Not Null |
| Country | Varchar(255) | Not Null |
| PhoneNumber | Varchar(255) | Unique, Not Null |

**3NF Review**: Each attribute depends solely on ContactID without any transitive relationships.

6. Managerial Hierarchy Table

|  |  |  |
| --- | --- | --- |
| Field | Type | Constraints |
| HierarchyID | INT | Primary Key, Auto Increment |
| ManagerID | INT | Not Null |
| SubordinateID | INT | Not Null |
| AssignmentDate | Date | Not Null |

**3NF Review**: ManagerID and SubordinateID might seem to introduce potential transitive dependencies, but since they are identifiers linking to EmployeeID, and the table is designed to track relationships rather than derive attributes from these relationships, it meets 3NF.

**7. Salary Table**

|  |  |  |
| --- | --- | --- |
| Field | Type | Constraints |
| SalaryID | INT | Primary Key, Auto Increment |
| EmployeeID | INT | Not Null |
| Amount | Decimal (10,2) | Not Null |
| EffectiveDate | Date | Not Null |
| SalaryType | Varchar(255) | Not Null |

**3NF Review**: Amount, EffectiveDate, and SalaryType are dependent only on SalaryID. This meets 3NF as there are no transitive dependencies.

**Employment Status Table -**

|  |  |  |
| --- | --- | --- |
| Field | Type | Constraints |
| StatusID | INT | Primary Key, Auto Increment |
| EmployeeID | INT | Not Null |
| Status | Varchar(255) | Not Null |
| StartDate | Date | Not Null |
| EndDate | Date | Null |

**3NF Review**: Each attribute in the Employment Status Table is directly dependent on the primary key (StatusID), with no transitive dependencies observed. The EmployeeID links this table to the Employee Table without introducing any dependencies between non-key attributes, ensuring the table meets Third Normal Form (3NF).

**Challenges Faced and Strategies to Overcome Them:**

**1. Challenge: Data Redundancy and Update Anomalies**

CA1 Issue: In the CA1 schema, I had fields like Address, City, Country directly within the Employee table, which led to data redundancy and complications in data management, especially in updating address details for multiple employees residing at the same address.

Strategy: In CA2, I refined the schema by extracting address-related information into a separate Address table with a unique AddressID. This normalization reduces redundancy by centralizing address information in one place, simplifying updates and ensuring consistency.

**2. Challenge: Handling Complex Relationships**

CA1 Issue: The CA1 schema managed foreign keys and relationships directly within the main tables, which might have made it difficult to maintain and visualize complex hierarchical and departmental relationships.

Strategy: In CA2, I introduced more defined relationship management through tables like the Managerial Hierarchy Table, which clearly delineates the reporting structures. This not only enhances the database's integrity but also its scalability by allowing for clear pathways to expand or modify managerial relationships.

**3. Challenge: Ensuring Flexible and Scalable Structure**

CA1 Issue: Initially, the job titles were likely embedded within the Employee table, limiting flexibility for role management and scalability as new job titles could require schema modifications.

Strategy: In CA2, you created a separate Jobs Table that houses job-related information, including language requirements and qualification levels. This modular approach enhances scalability (making it easier to add new jobs or modify existing ones) and reduces the load on the Employee table, focusing it solely on employee-specific data.

**4. Challenge: Improving Data Integrity and Accuracy**

CA1 Issue: Storing contact information directly in the Employee table might have led to issues with data integrity, especially with multiple contact numbers or emails per employee.

Strategy: CA2 introduces a Contact Table, segregating emails and phone numbers from the main employee records. This not only improves data integrity by ensuring unique and validated contact entries but also supports multiple contacts per employee, enhancing the system's functionality.

**5. Challenge: Data Accessibility and Management**

CA1 Issue: Managing employee statuses directly within the employee table could complicate queries, especially for historical data on employment status changes.

Strategy: The introduction of the EmploymentStatus Table in CA2 allows for a detailed timeline of employment statuses, enabling more straightforward historical data tracking and better management of employee status changes over time. This separation aids in complex queries and reporting needs.

**The rationale for your design decisions -**

**1. Separation of Concerns and Reduction of Redundancy**

From CA1 to CA2: Address details were moved from the Employee table to a dedicated Address table. This change adheres to the database normalization principle, specifically aimed at reducing redundancy and preventing anomalies during data updates.

Design Decision: By separating addresses into their own table and linking them via AddressID, updates to an address for multiple employees who reside at the same location become straightforward, minimizing data duplication and the potential for errors.

**2. Enhancement of Data Integrity and Accuracy**

From CA1 to CA2: Introduction of unique ContactID and Contact Table for managing contact details separate from the Employee table.

Design Decision: This separation enhances data integrity by ensuring that each contact detail can be uniquely identified and associated with an employee without redundancy. It supports scenarios where employees may have multiple email addresses or phone numbers, thus adhering to the principles of atomicity and 1NF.

**3. Improved Scalability and Flexibility**

From CA1 to CA2: Creation of discrete tables for Jobs and Departments to manage roles and departmental data separately from employee records.

Design Decision: This modular approach allows for easier modifications and additions to job titles or department structures without impacting the employee table structure. It supports scalability as the organization grows and roles evolve.

**4. Clear Definition of Hierarchical Relationships**

From CA1 to CA2: Implementation of a Managerial Hierarchy Table to explicitly define reporting structures.

Design Decision: This approach clarifies the relationships between managers and subordinates, essential for organizational hierarchy and reporting. It simplifies queries related to organizational structure and is critical for performance evaluations and management.

**5. Systematic Management of Employee Compensation**

From CA1 to CA2: Introduction of the Salary Table to manage detailed employee compensation data separately from the primary Employee table.

Design Decision: The Salary Table was created to store critical information regarding employee compensation, such as salary amounts, salary types and effective dates. This isolation from the Employee table ensures that changes in compensation are tracked meticulously without impacting core employee data.

**6. Robust Handling of Employment Status**

From CA1 to CA2: Establishing an EmploymentStatus Table that records changes in employment status over time.

Design Decision: By tracking the employment status with start and end dates in a dedicated table, the database can more accurately reflect and report on historical data, providing valuable insights into employment trends and patterns.