\*\*\*\*\*Back to ch3 to get the steps of design

\*\*\*\*\*alpha restaurant’s database points to the database name

Cover Page

**Summary**

**Table of Contents**

[**Chapter 1** Database Design processes: 1](#_Toc161065618)

[**1.1 Logical Design** 1](#_Toc161065619)

[**1.1.1 mission statement** 1](#_Toc161065620)

[**1.1.2 Mission Objectives** 1](#_Toc161065621)

[**1.1.3 Create Preliminary Field List** 2](#_Toc161065622)

[**1.1.4 Create preliminary table list** 2](#_Toc161065623)

[**1.1.5 Refining the Table Names** 3](#_Toc161065624)

[**1.1.6 Defining the Final Table List** 3](#_Toc161065625)

[**1.1.7 Composing the Table Descriptions** 3](#_Toc161065626)

[**1.1.8 Guidelines for Composing a Table Description** 3](#_Toc161065627)

[**1.1.9 the Final Table List** 3](#_Toc161065628)

[**1.1.10 Creating Field Names guidelines** 4](#_Toc161065629)

[**1.1.11 Associating Fields and Keys with Each Table** 4](#_Toc161065630)

[**1.1.12 Field Specifications** 8](#_Toc161065631)

[**1.1.13 Table Relationships** 18](#_Toc161065632)

[**1.1.14 Business Rules** 24](#_Toc161065633)

[**1.1.15 Views** 26](#_Toc161065634)

[**1.1.16 Index** 26](#_Toc161065635)

[**1.2 Physical implementation** 29](#_Toc161065636)

[**1.2.1 Database Creation** 29](#_Toc161065637)

# **Database Design processes:**

## Logical Design

The first phase involves determining and defining tables and their fields, establishing primary and foreign keys, establishing table relationships, and determining and establishing the various levels of data integrity.

### mission statement

The purpose of the Alpha Restaurant database is to maintain the data we generate and to supply information to the AI services that support of the entertainment services we provide to our clients and the data analysis tools support our sales and management business

### Mission Objectives

1. Maintain complete inventory information.
2. Maintain complete supplier information.
3. Maintain complete customer information
4. Maintain complete employees’ information
5. Maintain users chat messages
6. Track all customer sales.
7. Keep track of work orders
8. Manage rooms booking operation
9. Make sure every manager responsible for only one branch
10. Every employee can see only his data
11. Track the activities of customer friends
12. Support recommendation systems

### Create Preliminary Field List

Client first name order id booking id

Client last name customer id booking date

Client birthdate table id table id

Client number order date customer id

Client email order status start time

Client address order out date end time

Employee first name ingredient id item id

Employee last name ingredient name item name

Employee birthdate quantity in branch stock description

Employee number unit of measure preparation time

Employee email discount

Employee section branch id Item Status

Employee Branch item id branch id

Working start time order id

Working hours Item id attendance id

Item quantity employee id

table id Attendance time

capacity category id Time difference

table status category name

### Create preliminary table list

**Preliminary Table List**

Customers Booking Promotions Restaurant Sections Suppliers

Employees Ingredients Items Items Recipes Attendance

Storage Order Order Items Menu Categories Menu Items

Tables Branches Booking archive Accounts

### Refining the Table Names

1. Create a unique, descriptive name that is meaningful to the entire organization
2. Create a name that accurately, clearly, and unambiguously identifies the subject of the table.
3. Use the minimum number of words necessary to convey the subject of the table
4. Do not use words that convey physical characteristics. Avoid using words such as file, record, and table in the table name because they add a level of confusion that you don’t need.
5. Do not use acronyms and abbreviations. Acronyms are hard to decipher, abbreviations rarely convey the subject of the table, and both violate the first guideline in this list.
6. Do not use proper names or other words that will unduly restrict the data that can be entered into the table.
7. Do not use a name that implicitly or explicitly identifies more than one subject.
8. Using the plural form of the table name is a sound idea because it makes clear your intention to refer to a collection. Collections, of course, always take the plural (“Boats,” not “Boat”). In contrast, words that identify fields are always singular (“Home Phone,” not “Home Phones”).

### Defining the Final Table List

the list is “final” only in the sense that we’ve accounted for all the tables that we identified throughout the entire analysis process. It’s very likely that we’ll add new tables to this list based on requirements imposed by relationships, data integrity, or other information that we develop.

Recall that the four classifications we can use to identify the table type are **data, linking, subset, and validation**. There will be no linking tables or validation tables on the list because you have not yet defined relationships or imposed data integrity. The list will not contain subset tables because you define them after you assign fields to the data tables.

### Composing the Table Descriptions

The table description is another aspect of a table that you record on the Final Table List. A table description is crucial because it helps everyone understand why a given table exists and why the organization is concerned with collecting the data for that table. The table description also provides a means of validating the need for a table. If you are unable to explain why a table is important to the organization, then you need to determine when and how the table was identified and whether it is necessary at all.

### Guidelines for Composing a Table Description

1. Include a statement that accurately defines the table. Anyone should easily be able to determine the identity of the table from its description without any confusion or uncertainty.
2. Include a statement that explains why this table is important to the organization. A table contains data that is collected, maintained, manipulated, and retrieved by the organization for a particular reason
3. Do not include implementation-specific information in your table description, such as how or where the table is used. Avoid statements that indicate how you will specifically use this table, or how you will physically access it.
4. Do not make the table description for one table dependent upon the table description of another table. Each table description should be self-explanatory and independent from every other table description

### the Final Table List

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| Customers | Data | The users of our services offline or online ones, all customers information will be included to help in analyze customers needs and make sure our services Provided to wanted customers |
| Accounts | Data | Contain the sign in data of online services users |
| Employees | Data | The workers in all the sections and branches of the organization, keep their data to help in maintain sales and working flow |
| Suppliers | Data | Supplier information is vital because it allows us to maintain a constant supply of ingredients and ensure that our equipment is always in working order. |
| Tables | Data | Keep track of tables information which help us to manage Restaurant lounge and track usage of tables |
| Attendance | Data | Contain the attendance of employees in organization which help managers to track Workers' commitment |
| Storage | Data | Contain the quantity of all ingredients in the main storage which help Quartermaster and managers to manage ingredients orders |
| Restaurant Sections | validation | Contain a list of branch sections to help applying employees and orders to the right sections |
| Promotions | validation | Contain a list and description of every position in the working hierarchy |
| Branches | validation | Keep information of each branch to help us to track development of each branch work flow |
| Categories | validation | Contain the main titles of the menu to help branch to organize menu items under some of categories |
| Ingredients | Data | Keep track the ingredients which use in each branch and needing in stock for each one |
| Recipes | Data | To describe the item recipes in menu and to help in automate the report of Ingredients stock |
| Menu Items | Data | Contain the list of active and unactive items which is provided by branch |
| Booking | Data | Contain the info of the present and future reservations orders to the restaurant tables |
| Booking archive | subset | Contain the information of the past booking orders to help in track working flow of booking in branch |
| Order | Data | Contain the main orders details to help in provide order to the right customer and keep track of speed of working |
| Order Items | Link | Contain the specific details of each order to assign each item to the right section of restaurant and track speed of each section |

### Creating Field Names guidelines

1. Create a unique, descriptive name that is meaningful to the entire organization. A given field name should appear only once in the entire database; the only exception to this rule occurs when the field serves to establish a relationship between two tables.
2. Create a name that accurately, clearly, and unambiguously identifies the characteristic a field represents.
3. Use the minimum number of words necessary to convey the meaning of the characteristic the field represents. You want to avoid lengthy field names, but at the same time, you also want to avoid using a single word as a field name if that word is inappropriate.

### Associating Fields and Keys with Each Table

Keys are crucial to a table structure for the following reasons:

1. They ensure that each record in a table is precisely identified.
2. They help establish and enforce various types of integrity. they enable us to ensure that a table has unique records and that the fields you use to establish a relationship between a pair of tables always contain matching values.
3. They serve to establish table relationships.

The four main types of keys are candidate, primary, foreign, and non-keys. A key’s type determines its function within the table.

**Candidate Keys**

A field or set of fields that uniquely identifies a single instance (a record in the table) of the table’s subject. Each table must have at least one candidate key.

**Elements of a Candidate Key:**

1. It cannot be a multipart field.
2. It must contain unique values. This element helps us guard against duplicating a given record within the table.
3. It cannot contain Nulls. A Null value represents the absence of a value, and there’s absolutely no way a candidate key field can identify a given record if it is Null
4. Its value cannot cause a breach of the organization’s security or privacy rules. Values such as passwords and Social Security numbers are not suitable for use as a candidate key.
5. Its value is not optional in whole or in part. A value that is optional implies that it may be null at some point.
6. It comprises a minimum number of fields necessary to define uniqueness. You can use a combination of fields (treated as a single unit) to serve as a candidate key,
7. Its values must uniquely and exclusively identify each record in the table.
8. Its value can be modified only in rare or extreme cases.

**Artificial Candidate Keys (**surrogate**):**

When you determine that a table does not contain a candidate key, we can create and use an artificial (or surrogate) candidate key. we establish an artificial candidate key by creating a new field that conforms to all the Elements of a Candidate Key and then adding it to the table; this field becomes the official candidate key. we may also choose to create an artificial candidate key when it would be a stronger

**Primary Keys:**

1. A primary key field exclusively identifies the table throughout the database structure and helps establish relationships with other tables.
2. A primary key value uniquely identifies a given record within a table and exclusively represents that record throughout the entire database. It also helps to guard against duplicate records.

A primary key must conform to the exact same elements as a candidate key.

1. If we have a simple (single-field) candidate key and a composite candidate key, choose the simple candidate key. It’s always best to use a candidate key that contains the least number of fields.
2. We will Choose a candidate key that incorporates part of the table name within its own name.

**Elements of a Primary Key**

1. It cannot be a multipart field.
2. It must contain unique values.
3. It cannot contain Nulls.
4. Its value cannot cause a breach of the organization’s security or privacy rules.
5. Its value is not optional in whole or in part.
6. It comprises a minimum number of fields necessary to define uniqueness.
7. Its values must uniquely and exclusively identify each record in the table.
8. Its value must exclusively identify the value of each field within a given record.
9. Its value can be modified only in rare or extreme cases.

**Alternate Keys:**

Now that we’ve selected a candidate key to serve as the primary key for a particular table, we’ll designate the remaining candidate keys as alternate keys. These keys can be useful to us in an RDBMS program because they provide an alternative means of uniquely identifying a particular record within the table. we won’t be concerned with alternate keys for the remainder of the database design process, but we will work with them once again as you implement the database in an RDBMS program.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Customers | Accounts | Employees | Suppliers | branchTables | Branches |
| custID PK | custID FK | empID PK | suppID PK | tableID PR | branchID PK |
| custFName | custEmail | SSN AK | suppFName | branchID FK | branchAddress AK |
| custLName | custPassword | empFName | suppLName | tableStatus | managerID FK |
| custCity | CreatedDate | empLName | suppAddress | capacity |  |
| custStatus |  | empBirthDate | suppCompany |  |  |
| custBirthdate |  | empAddress |  |  |  |
| custAddress |  | empDateHired |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attendance | Positions | Promotions | Storage | Ingredients | Stock |
| AttendanceID PK | postionID PK | proID PK | storageID PK | ingredientsID PK | branchID CPK/FK |
| empID FK | postionName AK | previousPosID FK | ingredientsID FK | ingredientsName AK | ingredientsID CPK/FK |
| attendanceDate |  | newPosID FK | suppID FK | ingredientsUnit | quantity |
| DismissingTime |  |  |  |  |  |
| WorkedTime |  |  |  |  |  |
|  |  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Categories | branchMenu | BookingsArchive | Sections | Branch staff |
| categoryID PK | branchID FK | bookingID PK | sectionID PK | BranchID CPK/FK |
| categoryName AK | itemID FK | customerID FK | branchID FK | empID CPK/FK |
| categorySection FK | itemStatus | tableID FK | managerID FK | postionID FK |
| categoryDescription | discount | branchID FK |  | sectionID FK |
|  | price | bookingDate |  |  |
|  |  | startTime |  |  |
|  |  | endTime |  |  |
|  |  | startDate |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| MenuItems | BranchsMenu | Recipes | Orders | OrderItems | Bookings |
| itemID PK | branchID CPK/FK | recipeID PK | orderID PK | orderID CPK/FK | bookingID PK |
| itemName | itemID CPK/FK | ingredientsID FK | customerID FK | itemID CPK/FK | customerID FK |
| description |  | itemID fk | orderStatus | quantity | tableID FK |
| preparationTime |  | quantity | orderDate | quotePrice | branchID FK |
| categoryID FK |  | recipeStatus | shipDate |  | bookingDate |
| sectionID FK |  |  |  |  | startTime |
|  |  |  |  |  | endTime |
|  |  |  |  |  | startDate |

### Field Specifications

time it takes to establish field specifications for each field in the database is an investment toward building consistent data and quality information—we are not wasting time whatsoever by performing this process. In fact, we’ll waste more time in the end if you only partially perform this process or neglect it entirely.

**Field specifications are crucial for several reasons:**

1. Field specifications help establish and enforce field-level integrity.
2. Defining field specifications for each field enhances overall data integrity.
3. Defining field specifications compels we to acquire a complete understanding of the nature and purpose of the data in the database. Understanding the data means that we can judge whether the data is truly necessary and important to the organization, and we can learn how to use it to your best advantage.
4. Field specifications constitute the “data dictionary” of the database. Each field specification stores data on the characteristics of a particular field within the database.
5. This data dictionary is particularly useful when we implement your database in an RDBMS—we can use it as a guide for creating the fields and setting their fundamental properties.

**Anatomy of a Field Specification:**

A field specification incorporates various elements that define every attribute of a field. All the elements within the specification are categorized as general elements, physical elements, or logical elements.

Here are the elements within each category:

1. **General** **Elements**: Field Name, Parent Table, Specification Type, Source Specification, Shared By, Alias(es), Description
2. **Physical** **Elements**: Data Type, Length, Decimal Places, Character Support
3. **Logical** **Elements**: Key Type, Key Structure, Uniqueness, Null Support, Values Entered By, Required Value, Range of Values, Edit Rule
4. **General** **Elements**:
   1. **Field** **Name**: the set of absolute minimal words that uniquely identifies a particular field throughout the database.
   2. **parent** **table**: This is the only table in which the field will appear unless the field is participating in establishing a relationship.
   3. **Specification** **Type**: The elements you set for a given field depend upon the type of specification you define for the field. You can define a specification in three ways.
      1. **Unique**: This is the default specification for all fields except those that serve as a template for other fields or those that participate within a table relationship as foreign keys.
      2. **Generic**: This specification serves as a template for other field specifications and helps you ensure consistent definitions for fields that have the same general meaning.
      3. **Replica**: This is the default specification for a field based on a generic field or a field that serves as a foreign key within a table relationship, and it draws a majority of its element settings from an existing specification.
   4. **Source** **Specification**: is set only on a Replica specification and indicates the name of the specific field specification upon which the current specification is based.
   5. **Shared** **By** element indicates the names of other tables that share this field. The only table names that should appear here are those that have an explicit relationship to the field’s parent table.
   6. **Alias** is a name (or set of names) that you use for the field in very rare circumstances.
   7. **Description** is a complete interpretation of the field. Composing a field description is extremely beneficial because it forces you (and everyone in the organization) to think carefully about the nature of the data that will be stored in the field.
5. **Physical** **Elements**:
   1. **Data** **Type** element indicates the nature of the data that the field stores.
   2. Length element specifies the total number of characters that a user can enter for any given field value.
   3. **Decimal** **Places** element denotes the number of digits to the right of the decimal point in a real number.
   4. **Character** **Support** element indicates the type of characters that a user can enter into a given field value. Setting and enforcing this element helps you ensure that the user cannot introduce meaningless data into the field, thus enhancing field-level integrity
6. **Logical** **Elements**;
   1. **Key** **Type** element designates a field’s role within a table, which you identified as you were establishing a primary key for the table.
   2. **Key** **Structure** element denotes whether a field designated as a primary key is acting as a simple (single-field) primary key or as part of a composite (multifield) primary key
   3. **Uniqueness** element indicates whether a field’s values are unique. You set it as “Unique” when the Key Type element is set to “Primary”; otherwise, you’ll typically set this element as “non-unique.”
   4. **Null** **Support** specifies whether a field accepts Null.
   5. **Required** **Value** element denotes whether a user is required to enter a value for a field.
   6. **Range of Values** element specifies every possible valid value for a field.
   7. **Edit Rule** designates at what point a user can enter a value into a field and whether he can modify that value.

**Defining Field Specifications for Each Field in the Database:**

Field Specifications

|  |  |
| --- | --- |
| **Date Created:** ‏11‏/02‏/2024 | **Date Modified:** Click or tap to enter a date. |
| **Created By:** Ahmed Ismail | **Modified By:** Click or tap here to enter text. |
|  |  |

***General Elements***

|  |  |
| --- | --- |
| **Field Name:** custID | **Specification Type:**   Unique  Generic  Replica |
| **Parent Table:**  Customers | **Source Specification:** |
| **Shared By:** Booking, BookingArchive, OrderItems | |
| **Alias(es):** Click or tap here to enter text. | |
| **Description:** A unique number used to identify each customer in our organization, whether it is a offline agent or online. It is assigned when a customer place they first order | |

***Physical Elements***

|  |  |
| --- | --- |
| **Data Type:** serial | **Character Support:**  Letters (A-Z)  Keyboard (. , / $ # %)  Numbers (0-9)  Special (© ® ™ £ π) |
| **Length:** 1 to 10 |
| **Decimal Places:** 0 |
| **Input Mask:** Click or tap here to enter text. | |
| **Display Format:** Click or tap here to enter text. | |

***Logical Elements***

|  |  |
| --- | --- |
| **Key Type:**   Non  Primary  Foreign  Alternate | **Edit Rule:**  Enter Now, Edits Allowed  Enter Now, Edits Not Allowed  Enter Later, Edits Allowed  Enter Later, Edits Not Allowed  Not Determined at This Time |
| **Key Structure:**   Simple  Composite |
| **Uniqueness:**   non-Unique  Unique |
| **Null Support:**  Nulls Allowed  No Nulls |
| **Values Entered By:** User  System |
| **Required Value:**  No Yes |
| **Default Value:** Incremental |
| **Range of Values:** Click or tap here to enter text. | |
| **Comparisons Allowed:**  Same Field  All  =  >  >=  ≠  <  <=  Other Fields  All  =  >  >=  ≠  <  <=  Value Expression  All  =  >  >=  ≠  <  <= | |

Field Specifications

|  |  |
| --- | --- |
| **Date Created:** ‏11‏/02‏/2024 | **Date Modified:** Click or tap to enter a date. |
| **Created By:** Ahmed Ismail | **Modified By:** Click or tap here to enter text. |
|  |  |

***General Elements***

|  |  |
| --- | --- |
| **Field Name:** custFName | **Specification Type:**   Unique  Generic  Replica |
| **Parent Table:** customers | **Source Specification:** |
| **Label:** Click or tap here to enter text. |
| **Shared By:** Click or tap here to enter text. | |
| **Alias(es):** Click or tap here to enter text. | |
| **Description:** A string to identify the first name of each customer | |

***Physical Elements***

|  |  |
| --- | --- |
| **Data Type:** character | **Character Support:**  Letters (A-Z)  Keyboard (. , / $ # %)  Numbers (0-9)  Special (© ® ™ £ π) |
| **Length:** 20 |
| **Decimal Places:** Click or tap here to enter text. |
| **Input Mask:** Click or tap here to enter text. | |
| **Display Format:** Text | |

***Logical Elements***

|  |  |
| --- | --- |
| **Key Type:**   Non  Primary  Foreign  Alternate | **Edit Rule:**  Enter Now, Edits Allowed  Enter Now, Edits Not Allowed  Enter Later, Edits Allowed  Enter Later, Edits Not Allowed  Not Determined at This Time |
| **Key Structure:**   Simple  Composite |
| **Uniqueness:**   non-Unique  Unique |
| **Null Support:**  Nulls Allowed  No Nulls |
| **Values Entered By:**  User  System |
| **Required Value:**  No Yes |
| **Default Value:** Click or tap here to enter text. |
| **Range of Values:** Click or tap here to enter text. | |
| **Comparisons Allowed:**  Same Field  All  =  >  >=  ≠  <  <=  Other Fields  All  =  >  >=  ≠  <  <=  Value Expression  All  =  >  >=  ≠  <  <= | |

Field Specifications

|  |  |
| --- | --- |
| **Date Created:** ‏11‏/02‏/2024 | **Date Modified:** Click or tap to enter a date. |
| **Created By:** Ahmed Ismail | **Modified By:** Click or tap here to enter text. |
|  |  |

***General Elements***

|  |  |
| --- | --- |
| **Field Name:** custLName | **Specification Type:**   Unique  Generic  Replica |
| **Parent Table:** customers | **Source Specification:** |
| **Label:** Click or tap here to enter text. |
| **Shared By:** Click or tap here to enter text. | |
| **Alias(es):** Click or tap here to enter text. | |
| **Description:** A string to identify the first name of each customer | |

***Physical Elements***

|  |  |
| --- | --- |
| **Data Type:** character | **Character Support:**  Letters (A-Z)  Keyboard (. , / $ # %)  Numbers (0-9)  Special (© ® ™ £ π) |
| **Length:** 20 |
| **Decimal Places:** Click or tap here to enter text. |
| **Input Mask:** Click or tap here to enter text. | |
| **Display Format:** Text | |

***Logical Elements***

|  |  |
| --- | --- |
| **Key Type:**   Non  Primary  Foreign  Alternate | **Edit Rule:**  Enter Now, Edits Allowed  Enter Now, Edits Not Allowed  Enter Later, Edits Allowed  Enter Later, Edits Not Allowed  Not Determined at This Time |
| **Key Structure:**   Simple  Composite |
| **Uniqueness:**   non-Unique  Unique |
| **Null Support:**  Nulls Allowed  No Nulls |
| **Values Entered By:**  User  System |
| **Required Value:**  No Yes |
| **Default Value:** Click or tap here to enter text. |
| **Range of Values:** Click or tap here to enter text. | |
| **Comparisons Allowed:**  Same Field  All  =  >  >=  ≠  <  <=  Other Fields  All  =  >  >=  ≠  <  <=  Value Expression  All  =  >  >=  ≠  <  <= | |

Field Specifications

|  |  |
| --- | --- |
| **Date Created:** ‏11‏/02‏/2024 | **Date Modified:** Click or tap to enter a date. |
| **Created By:** Ahmed Ismail | **Modified By:** Click or tap here to enter text. |
|  |  |

***General Elements***

|  |  |
| --- | --- |
| **Field Name:** custCity | **Specification Type:**   Unique  Generic  Replica |
| **Parent Table:**  customers | **Source Specification:** |
| **Label:** Click or tap here to enter text. |
| **Shared By:** Click or tap here to enter text. | |
| **Alias(es):** Click or tap here to enter text. | |
| **Description:** a identifier for the Living city of each customer | |

***Physical Elements***

|  |  |
| --- | --- |
| **Data Type:** character | **Character Support:**  Letters (A-Z)  Keyboard (. , / $ # %)  Numbers (0-9)  Special (© ® ™ £ π) |
| **Length:** 35 |
| **Decimal Places:** Click or tap here to enter text. |
| **Input Mask:** Click or tap here to enter text. | |
| **Display Format:** Click or tap here to enter text. | |

***Logical Elements***

|  |  |
| --- | --- |
| **Key Type:**   Non  Primary  Foreign  Alternate | **Edit Rule:**  Enter Now, Edits Allowed  Enter Now, Edits Not Allowed  Enter Later, Edits Allowed  Enter Later, Edits Not Allowed  Not Determined at This Time |
| **Key Structure:**   Simple  Composite |
| **Uniqueness:**   non-Unique  Unique |
| **Null Support:**  Nulls Allowed  No Nulls |
| **Values Entered By:** User  System |
| **Required Value:**  No Yes |
| **Default Value:** Click or tap here to enter text. |
| **Range of Values:** in CITIES Table | |
| **Comparisons Allowed:**  Same Field  All  =  >  >=  ≠  <  <=  Other Fields  All  =  >  >=  ≠  <  <=  Value Expression  All  =  >  >=  ≠  <  <= | |

Field Specifications

|  |  |
| --- | --- |
| **Date Created:** ‏11‏/02‏/2024 | **Date Modified:** Click or tap to enter a date. |
| **Created By:** Ahmed Ismail | **Modified By:** Click or tap here to enter text. |
|  |  |

***General Elements***

|  |  |
| --- | --- |
| **Field Name:** custAddress | **Specification Type:**   Unique  Generic  Replica |
| **Parent Table:** customers | **Source Specification:** |
| **Label:** Click or tap here to enter text. |
| **Shared By:** Click or tap here to enter text. | |
| **Alias(es):** Click or tap here to enter text. | |
| **Description:** the living address for each customer not necessarily the main address of orders | |

***Physical Elements***

|  |  |
| --- | --- |
| **Data Type:** character | **Character Support:**  Letters (A-Z)  Keyboard (. , / $ # %)  Numbers (0-9)  Special (© ® ™ £ π) |
| **Length:** Max 95 |
| **Decimal Places:** Click or tap here to enter text. |
| **Input Mask:** Click or tap here to enter text. | |
| **Display Format:** Click or tap here to enter text. | |

***Logical Elements***

|  |  |
| --- | --- |
| **Key Type:**   Non  Primary  Foreign  Alternate | **Edit Rule:**  Enter Now, Edits Allowed  Enter Now, Edits Not Allowed  Enter Later, Edits Allowed  Enter Later, Edits Not Allowed  Not Determined at This Time |
| **Key Structure:**   Simple  Composite |
| **Uniqueness:**   non-Unique  Unique |
| **Null Support:**  Nulls Allowed  No Nulls |
| **Values Entered By:** User  System |
| **Required Value:**  No Yes |
| **Default Value:** Click or tap here to enter text. |
| **Range of Values:** Click or tap here to enter text. | |
| **Comparisons Allowed:**  Same Field  All  =  >  >=  ≠  <  <=  Other Fields  All  =  >  >=  ≠  <  <=  Value Expression  All  =  >  >=  ≠  <  <= | |

Field Specifications

|  |  |
| --- | --- |
| **Date Created:** ‏11‏/02‏/2024 | **Date Modified:** Click or tap to enter a date. |
| **Created By:** Ahmed Ismail | **Modified By:** Click or tap here to enter text. |
|  |  |

***General Elements***

|  |  |
| --- | --- |
| **Field Name:** custStatus | **Specification Type:**   Unique  Generic  Replica |
| **Parent Table:**  customers | **Source Specification:** |
| **Label:** Click or tap here to enter text. |
| **Shared By:** Click or tap here to enter text. | |
| **Alias(es):** Click or tap here to enter text. | |
| **Description:** an identifier for the Living city of each customer | |

***Physical Elements***

|  |  |
| --- | --- |
| **Data Type:** character | **Character Support:**  Letters (A-Z)  Keyboard (. , / $ # %)  Numbers (0-9)  Special (© ® ™ £ π) |
| **Length:** 35 |
| **Decimal Places:** Click or tap here to enter text. |
| **Input Mask:** Click or tap here to enter text. | |
| **Display Format:** Click or tap here to enter text. | |

***Logical Elements***

|  |  |
| --- | --- |
| **Key Type:**   Non  Primary  Foreign  Alternate | **Edit Rule:**  Enter Now, Edits Allowed  Enter Now, Edits Not Allowed  Enter Later, Edits Allowed  Enter Later, Edits Not Allowed  Not Determined at This Time |
| **Key Structure:**   Simple  Composite |
| **Uniqueness:**   non-Unique  Unique |
| **Null Support:**  Nulls Allowed  No Nulls |
| **Values Entered By:** User  System |
| **Required Value:**  No Yes |
| **Default Value:** Click or tap here to enter text. |
| **Range of Values:** in CITIES Table | |
| **Comparisons Allowed:**  Same Field  All  =  >  >=  ≠  <  <=  Other Fields  All  =  >  >=  ≠  <  <=  Value Expression  All  =  >  >=  ≠  <  <= | |

Field Specifications

|  |  |
| --- | --- |
| **Date Created:** ‏11‏/02‏/2024 | **Date Modified:** Click or tap to enter a date. |
| **Created By:** Ahmed Ismaill | **Modified By:** Click or tap here to enter text. |
|  |  |

***General Elements***

|  |  |
| --- | --- |
| **Field Name:** custBirthdate | **Specification Type:**   Unique  Generic  Replica |
| **Parent Table:** customers | **Source Specification:** |
| **Label:** Click or tap here to enter text. |
| **Shared By:** Click or tap here to enter text. | |
| **Alias(es):** Click or tap here to enter text. | |
| **Description:** The Birthdate for each customer | |

***Physical Elements***

|  |  |
| --- | --- |
| **Data Type:** Date | **Character Support:**  Letters (A-Z)  Keyboard (. , / $ # %)  Numbers (0-9)  Special (© ® ™ £ π) |
| **Length:** Click or tap here to enter text. |
| **Decimal Places:** Click or tap here to enter text. |
| **Input Mask:** Click or tap here to enter text. | |
| **Display Format:** Click or tap here to enter text. | |

***Logical Elements***

|  |  |
| --- | --- |
| **Key Type:**   Non  Primary  Foreign  Alternate | **Edit Rule:**  Enter Now, Edits Allowed  Enter Now, Edits Not Allowed  Enter Later, Edits Allowed  Enter Later, Edits Not Allowed  Not Determined at This Time |
| **Key Structure:**   Simple  Composite |
| **Uniqueness:**   non-Unique  Unique |
| **Null Support:**  Nulls Allowed  No Nulls |
| **Values Entered By:** User  System |
| **Required Value:**  No Yes |
| **Default Value:** Click or tap here to enter text. |
| **Range of Values:** Click or tap here to enter text. | |
| **Comparisons Allowed:**  Same Field  All  =  >  >=  ≠  <  <=  Other Fields  All  =  >  >=  ≠  <  <=  Value Expression  All  =  >  >=  ≠  <  <= | |

### Table Relationships

a relationship exists between two tables when you can, in some way, associate the records of the first table with those of the second.

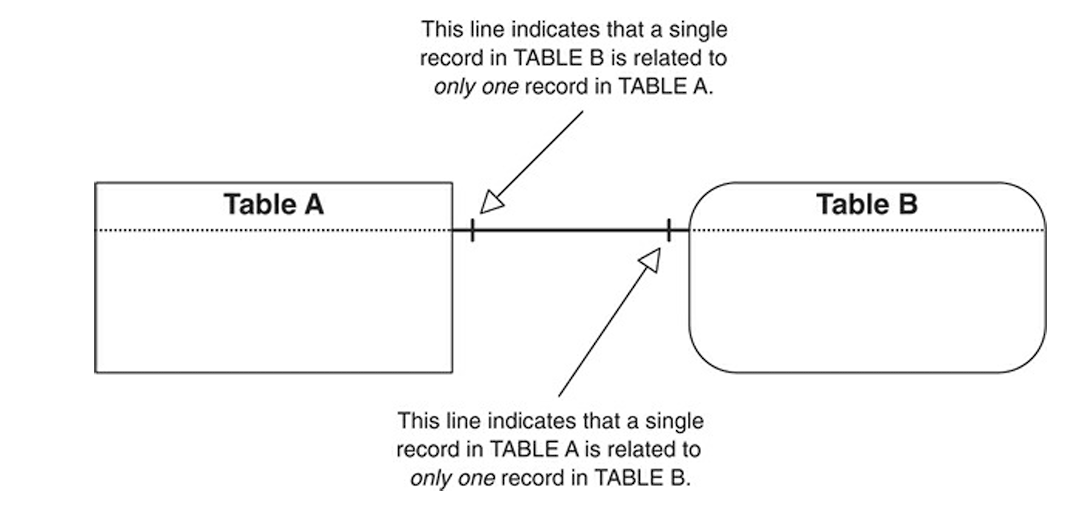
A relationship is an important component of a relational database.

1. It establishes a connection between a pair of tables that are logically related to each other. A pair of tables is logically related via the data each contains.
2. It helps to further refine table structures and minimize redundant data. As you establish a relationship between a pair of tables, you will inevitably make minor modifications to the table structures.
3. It is the mechanism that enables you to draw data from multiple tables simultaneously.
4. A properly defined relationship ensures relationship-level integrity, which guarantees that the relationship itself is reliable and sound.

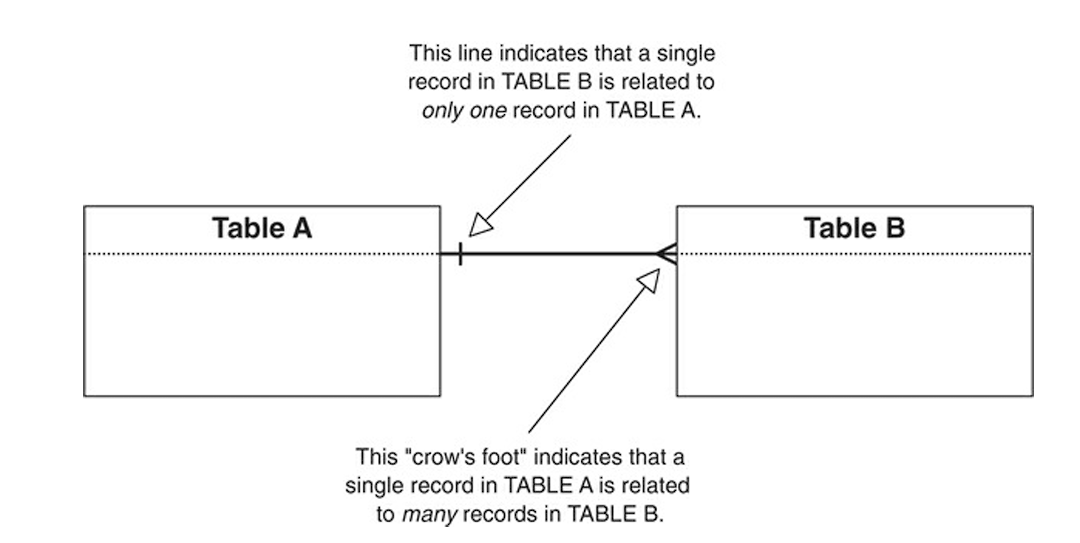
**Types of Relationships:**

Three specific types of relationships can exist between a pair of tables: one-to-one, one-to-many, and many-to-many.

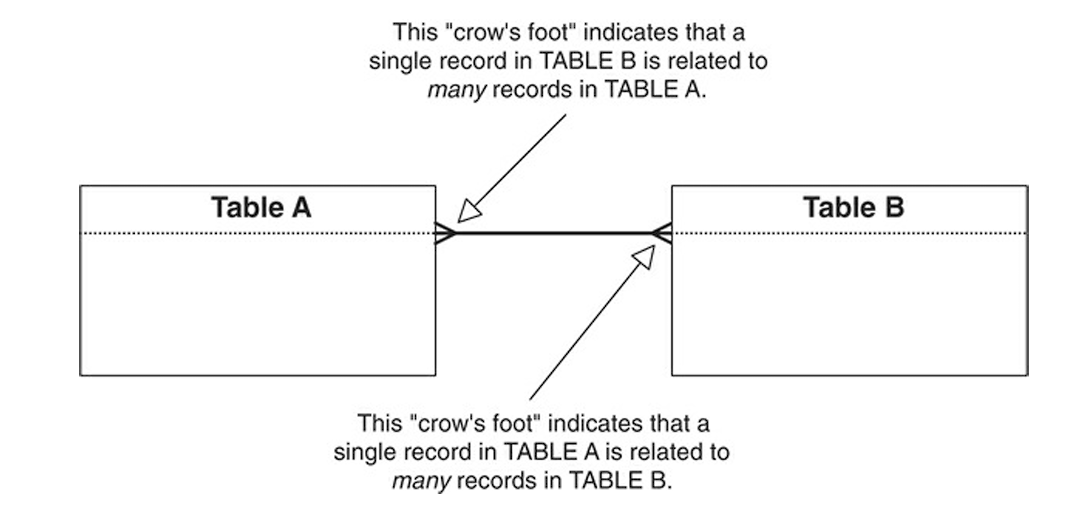
1. **one-to-one relationship** when a single record in the first table is related to only one record in the second table, and a single record in the second table is related to only one record in the first table.



1. **A one-to-many** **relationship** exists between a pair of tables when a single record in the first table can be related to one or more records in the second table, but a single record in the second table can be related to only one record in the first table.



1. **many-to-many relationship** when a single record in the first table can be related to one or more records in the second table and a single record in the second table can be related to one or more records in the first table.



We will Begin the process of identifying relationships by creating a matrix of all the tables in your database.

As we work with a pair of tables, ask the participants questions about the records in each table. our goal is to determine the relationship between a single record in one table to one or more records in the other table, and vice versa.

The two types of questions you can ask are:

1. **Associative**: This is a simple and straightforward type of question that you can generically phrase as follows: Can a single record in (name of first table) be associated with one or more records in (name of second table)?
2. **Contextual**: This type of question contrasts a single instance of the subject represented by the first table against multiple instances of the subject represented by the second table.
   1. **Ownership-oriented** questions include words or phrases such as own, has, is part of, and contain.
   2. **Action-oriented** questions incorporate action verbs such as make, visit, place, teach, and attend.

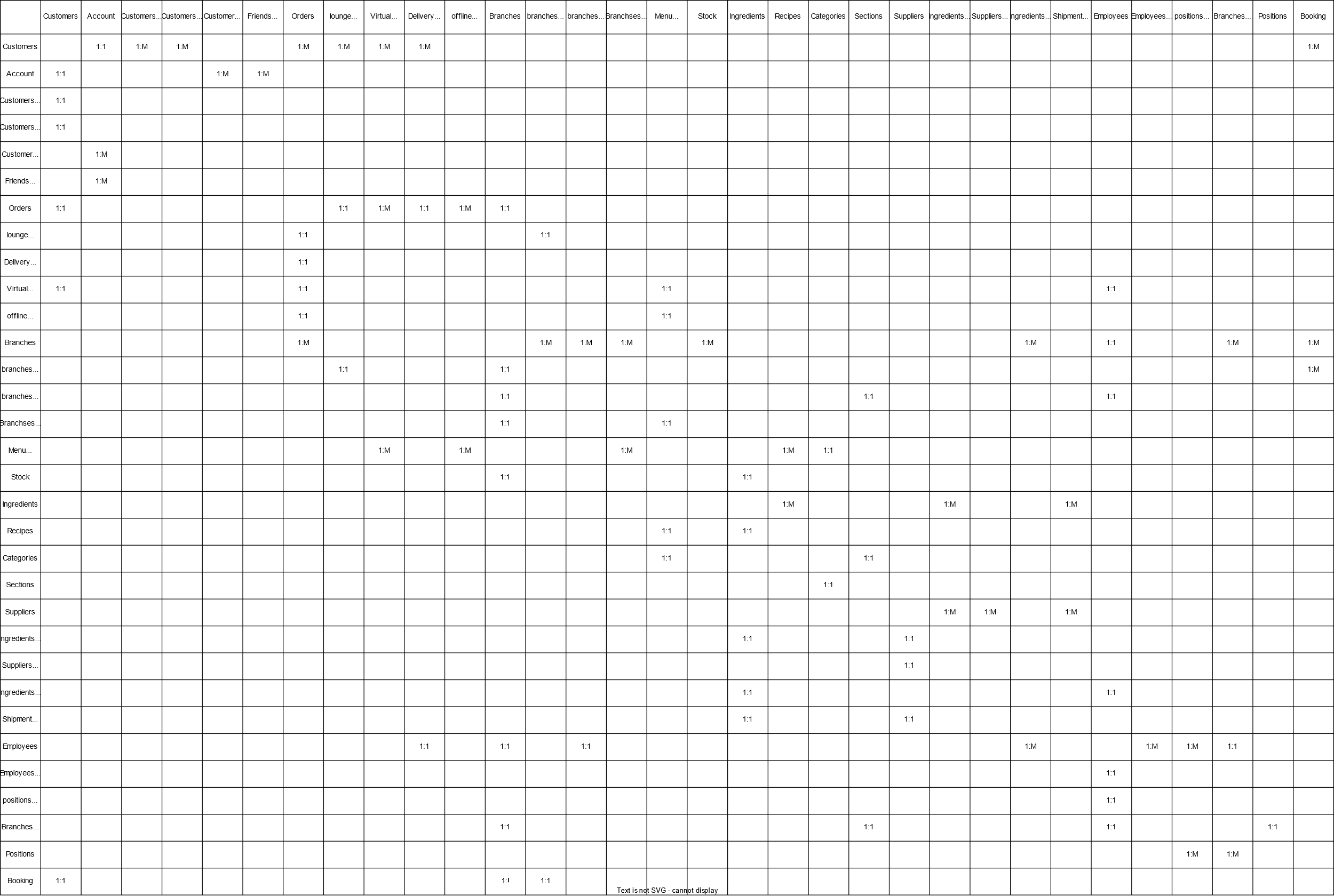
As you work down the list of tables in the matrix, you’ll eventually realize that you’re asking questions about a given pair of tables twice—once from the perspective of the first table and then again from the perspective of the second table. The answers to both of these questions will identify the type of relationship that exists between the tables.

Relationships will often differ from one perspective to the other, and we must know how to determine what type of relationship officially exists between each pair of tables on the matrix. You make this determination using the following set of formulas; each formula corresponds to a particular relationship-type definition.

**1:1 + 1:1 = 1:1** A pair of tables bears a one-to-one relationship when a single record in the first table is related to only one record in the second table, and a single record in the second table is related to only one record in the first table.

**1:N + 1:1 = 1:N** A one-to-many relationship exists between a pair of tables when a single record in the first table can be related to one or more records in the second table, but a single record in the second table can be related to only one record in the first table.

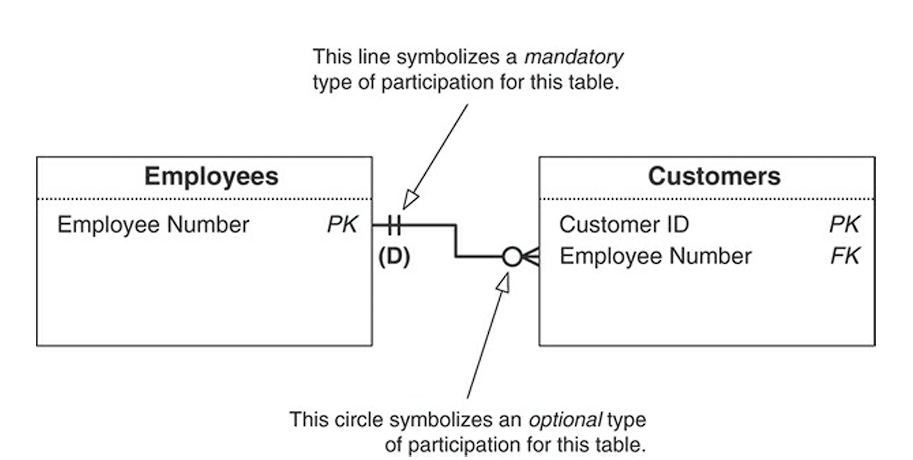
**1:N + 1:N = M:N** A pair of tables bears a many-to-many relationship when a single record in the first table can be related to one or more records in the second table and a single record in the second table can be related to one or more records in the first table.



**Identifying the Type of Participation for Each Table:**

The type of participation we assign to a given table determines whether a record must exist in that table before you can enter records into the related table. The two types of participation are:

1. **Mandatory**: At least one record must exist in this table before you can enter any records into the related table.
2. **Optional**: There is no requirement for any records to exist in this table before you can enter records into the related table.



**ERD:**

**Defining a Deletion Rule for Each Relationship:**

Deletion rules are crucial to relationship-level integrity because they help guard against orphaned records, which are records in the child table that have no relationship whatsoever to any records in the parent table.

These are the five types of deletion rules you can define and the actions the RDBMS should take when a given rule is in force:

1. **Deny**: The RDBMS will not delete the record in the parent table, but will instead keep the record and designate it as “inactive.”

2. **Restrict**: The RDBMS will not delete the record in the parent table if related records exist in the child table. we must have the RDBMS delete all the related records in the child table before you can have it delete the record in the parent table.

3. **Cascade**: The RDBMS will take two specific actions: It will delete the record in the parent table, and it will also automatically delete all related records in the child table.

4. **Nullify**: The RDBMS will delete the record in the parent table and will then update the foreign key values of related records in the child table to Null. If we are going to use this deletion rule, we must modify the foreign key’s field specifications and set the Null Support logical element to “Nulls Allowed.”

5. **Set** **Default**: The RDBMS will delete the record in the parent table and will then update the foreign key values of related records in the child table to a specified default value you’ve set in the RDBMS. Obviously, we must have a setting for a default value to use this rule.

We Use (D) for Deny, (R) for Restrict, (C) for Cascade, (N) for Nullify, and (S) for Set Default. We will Place the designation under the connection line of the parent table.

we always set the deletion rule from the perspective of the parent table because it is the more important of the two tables within the relationship. Deleting a record in the parent table will always have some effect on related records in the child table, but deleting a record in the child table will have little if any effect on the related record in the parent table.

**Identifying the Degree of Participation for Each Table:**

The degree of participation indicates the minimum number of records that a given table must have associated with a single record in the related table and the maximum number of records that the table is allowed to have associated with a single record in the related table. The factors we use to determine the degree of participation obvious circumstances, common sense, or conformance to some set of standards are the same as those you used to determine the type of participation.

You use two numbers separated by a comma and enclosed within parentheses to represent the degree of participation for a given table. The first number indicates the required minimum number of related records, and the second number indicates the allowable maximum number of related records.

### Business Rules

A business rule is a statement that imposes some form of constraint on a specific aspect of the database, such as the elements within a field specification for a particular field or the characteristics of a given relationship.

Creating a generic set of business rules that could apply to two or more organizations is next to impossible. Each organization has its own data and information requirements, and each has its own unique way of conducting its business; therefore, every organization needs its own specific set of business rules.

**Types of Business Rule:**

The two major types of business rules are database oriented and application oriented. Both types of business rules impose some form of constraint and help enforce and maintain overall data integrity, but they differ with regard to where and how they are established.

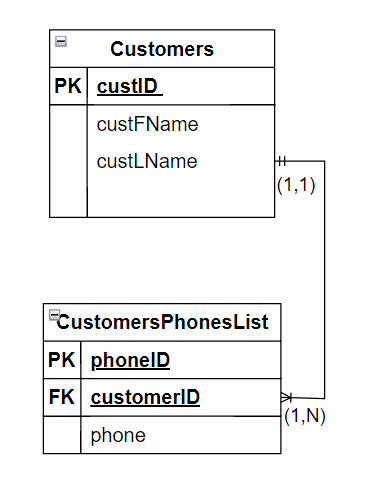
1. **Database-oriented business** rules impose constraints that you can establish within the logical design of the database. You implement a given constraint by modifying various field specification elements, relationship characteristics, or a combination of the two. The statement from which you derive the constraint is a database-oriented business rule if you can meaningfully and clearly establish the constraint by either of these means.
2. **Application-oriented** business rules impose constraints that you cannot establish within the logical design of the database. You must instead establish them within the physical design of the database or within the design of a database application, where they will be more applicable and meaningful.

**Field-Specific Business Rules**

Business rules under the field-specific category impose constraints on the elements of a field specification for a particular field. The number of elements a given rule affects depends on the manner in which you define that rule. For example, this rule only affects one element: Order dates cannot be earlier than May 20, 2024, which is the date our business was formed

**Relationship-Specific Business Rules**

Relationship-specific business rules impose constraints that affect the characteristics of a relationship.



Say you determine that there must be a limit to the number of phones for each customer, and you define the following business rule: Each customer must have a minimum of 1 number, but and unlimited maximum

**Business Rule Specifications Sheets:**

The Business Rule Specifications sheet contains the following items.

1. **Statement**: This is the text of the business rule itself. It should be clear and succinct and should convey the required constraints without any confusion or ambiguity.
2. **Constraint**: This is a brief explanation of how the constraint applies to the tables and fields. For instance, we can use the following explanation for the constraint imposed by the business rule in the preceding
3. **Type**: Here is where we indicate whether the rule is database oriented or application oriented.
4. **Category**: This is where we indicate whether the rule is field specific or relationship specific.
5. **Test** **On**: Here is where we indicate which actions (insert, delete, update) will test the constraint the business rule imposes.
6. **Structures** **Affected**: Depending on the type of business rule, the constraint will affect either a field or a relationship. This is where you designate the name of the field(s) the rule will affect or the name of the table(s) involved in the relationship that the rule affects.
7. **Field** **Elements** **Affected**: A business rule that pertains to a field can affect one or more elements of that field’s specifications. This is where you indicate the elements the rule affects.
8. **Relationship** **Characteristics** **Affected**: A business rule that pertains to a relationship will affect one or more of the relationship’s characteristics.
9. **Action** **Taken**: Here we indicate the modifications you’ve made to the elements of a field specification or to a relationship diagram. Begin our entry with the date the modification was made and the name or initials of the person who authorized or made the change.

### Views

## Physical implementation

The second phase entails creating the tables, establishing key fields and table relationships, and using the proper tools to implement the various levels of data integrity.

### Database Creation

So, we successfully installed PostgreSQL; we should have one superuser, whose password we know by heart. Now you should take the following steps to set up additional roles and assign privileges:

1. PostgreSQL creates one superuser and one database for you at installation, both named postgres. Log in to our server as postgres.

2. Before creating our first database, create a role that will own the database and can log in, such as:

CREATE ROLE mydb\_admin LOGIN PASSWORD 'something';

3. Create the database and set the owner:

CREATE DATABASE mydb WITH owner = mydb\_admin;

4. Now log in as the mydb\_admin user and start setting up additional schemas and tables.

The minimum SQL command to create a database is: CREATE DATABASE alpha;

**Using Schemas:**

Schemas organize our database into logical groups. If we have more than two dozen tables in our database, consider cubby holing them into schemas. Objects must have unique names within a schema but need not be unique across the database. If we cram all our tables into the default public schema, we’ll run into name clashes sooner or later.

CREATE SCHEMA customers;

### Index

Indexes are special tables that, unlike normal data tables, are kept in a specific order. Instead of containing all of the data about an entity, however, an index contains only the column (or columns) used to locate rows in the data table, along with information describing where the rows are physically located. Therefore, the role of indexes is to facilitate the retrieval of a subset of a table’s rows and columns without the need to inspect every row in the table.

the system would have to scan the entire table, row by row, to find all matching entries. If there are many rows in test1 and only a few rows (perhaps zero or one) that would be returned by such a query, this is clearly an inefficient method. But if the system has been instructed to maintain an index on the id column, it can use a more efficient method for locating matching rows. For instance, it might only have to walk a few levels deep into a search tree.

A similar approach is used in most non-fiction books: terms and concepts that are frequently looked up by readers are collected in an alphabetic index at the end of the book. The interested reader can scan the index relatively quickly and flip to the appropriate page(s), rather than having to read the entire book to find the material of interest. Just as it is the task of the author to anticipate the items that readers are likely to look up, it is the task of the database programmer to foresee which indexes will be useful.

**Index Types in PostgreSQL:**

PostgreSQL provides several index types: B-tree, Hash, GiST, SP-GiST, GIN and BRIN. Each index type uses a different algorithm that is best suited to different types of indexable clauses. By default, the CREATE INDEX command creates B-tree indexes, which fit the most common situations. The other index types are selected by writing the keyword USING followed by the index type name. we will use only few types of those types:

1. B-tree: B-Tree index (the B stands for Balanced). B-Tree indexes are the most common and beneficial data structure across relational database management systems (RDBMSes). There are two objectives behind a B-Tree index. The first and primary goal is to enable finding records rapidly and efficiently instead of having to perform a sequential table scan. The second ancillary goal is to enable the quick sorting of data. To achieve both these goals, a B-Tree index stores the data it contains in sorted order and has a search tree.

B-trees can handle equality and range queries on data that can be sorted into some ordering. In particular, the PostgreSQL query planner will consider using a B-tree index whenever an indexed column is involved in a comparison using one of these operators: < <= = >= >

1. Hash: A hash index implements a variation of a hash table data structure where a hashing function takes the index key value, produces a 4-byte signed int value (32-bit) representing the key value, and stores the hashed value in something known as a bucket along with a pointer to where the record exists in the heap table (essentially a set of tuples). The query planner will consider using a hash index whenever an indexed column is involved in a comparison using the equal operator: =

Index Sets:

1. **Primary Index**:
   * A primary index is an index defined on a set of fields that includes the primary key of a table. It's typically used for efficient retrieval of records based on their primary key values.
2. **Secondary Index**:
   * A secondary index is an index created on a set of fields other than the primary key. It's used for efficient retrieval of records based on non-primary key values.
3. **Unique Index**:
   * A unique index is an index that enforces uniqueness on the indexed column(s), similar to a primary key. However, unlike a primary key, it allows null values.
4. **Composite Index**:
   * a composite index is created on multiple columns. It allows efficient retrieval of records based on combinations of those columns.

Field Specifications

|  |  |
| --- | --- |
| **Date Created:** Click here to enter a date. | **Date Modified:** Click or tap to enter a date. |
| **Created By:** Click or tap here to enter text. | **Modified By:** Click or tap here to enter text. |
|  |  |

***General Elements***

|  |  |
| --- | --- |
| **Field Name:** Click or tap here to enter text. | **Specification Type:**   Unique  Generic  Replica |
| **Parent Table:** Click or tap here to enter text. | **Source Specification:** |
| **Label:** Click or tap here to enter text. |
| **Shared By:** Click or tap here to enter text. | |
| **Alias(es):** Click or tap here to enter text. | |
| **Description:** Click or tap here to enter text. | |

***Physical Elements***

|  |  |
| --- | --- |
| **Data Type:** Click or tap here to enter text. | **Character Support:**  Letters (A-Z)  Keyboard (. , / $ # %)  Numbers (0-9)  Special (© ® ™ £ π) |
| **Length:** Click or tap here to enter text. |
| **Decimal Places:** Click or tap here to enter text. |
| **Input Mask:** Click or tap here to enter text. | |
| **Display Format:** Click or tap here to enter text. | |

***Logical Elements***

|  |  |
| --- | --- |
| **Key Type:**   Non  Primary  Foreign  Alternate | **Edit Rule:**  Enter Now, Edits Allowed  Enter Now, Edits Not Allowed  Enter Later, Edits Allowed  Enter Later, Edits Not Allowed  Not Determined at This Time |
| **Key Structure:**   Simple  Composite |
| **Uniqueness:**   non-Unique  Unique |
| **Null Support:**  Nulls Allowed  No Nulls |
| **Values Entered By:** User  System |
| **Required Value:**  No Yes |
| **Default Value:** Click or tap here to enter text. |
| **Range of Values:** Click or tap here to enter text. | |
| **Comparisons Allowed:**  Same Field  All  =  >  >=  ≠  <  <=  Other Fields  All  =  >  >=  ≠  <  <=  Value Expression  All  =  >  >=  ≠  <  <= | |

Business Rule Specifications

***Rule Information***

|  |  |  |
| --- | --- | --- |
| **Statement:** Click or tap here to enter text. | | |
| **Constraints:** Click or tap here to enter text. | | |
| **Type:**  Database Oriented  Application Oriented | **Category:**   Field Specific  Relationship Specific | **Test On:**   Insert  Delete  Update |

***Structures Affected***

|  |
| --- |
| **Field Names:** Click or tap here to enter text. |
| **Tables Names:** Click or tap here to enter text. |

***Field Elements Affected***

|  |
| --- |
| **Physical Elements:**  Data Type  Length  Character Support |
| **Logical Elements:**  Key Type  Null Support  Range of values  Key Structure  Values Entered By  Edit Rule  Uniqueness  Required Value |

***Relationship Characteristic Affected***

|  |
| --- |
| **Physical Elements:**  Deletion Rule  Type of Participation  Degree of Participation |

***Action Taken***

|  |
| --- |
| Click or tap here to enter text. |

View Specifications

***General Information***

|  |  |
| --- | --- |
| **Name:** Click or tap here to enter text. | **Type:**   Data  Aggregate  Validation |
| **Description:** Click or tap here to enter text. | |

***Base Tables***

|  |
| --- |
| Click or tap here to enter text. |

***Calculated Elements Expressions***

|  |  |
| --- | --- |
| Field Name | Expression |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

***Filters***

|  |  |
| --- | --- |
| Field Name | Condition |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Jobs**

|  |  |  |  |
| --- | --- | --- | --- |
| job section | Job title | Description | Permissions |
| Restaurant | Branch Manager | * responsible for hiring applicants, letting employees go, training new hires, overseeing general restaurant activities, and working on marketing and community outreach strategies. * They may also help to set menu prices and purchase supplies. | Insert work days, show branch tables view, show stock details, give permission for stock orders, -- Image processing, |
| Assistance Manager |  |  |
| Kitchen manager | A kitchen manager helps manage the back of house team, including prep and clean up. They help ensure all sanitation standards are met. They often are responsible for ordering ingredients and replacing or adding equipment within a budget. |  |
| Cashier | The cashier runs the cash register, processes payments, and interacts with customers. Sometimes a cashier also takes orders from guests. |  |
| Chief | * Responsible for planning menus, preparing and cooking food, and maintaining food quality and presentation. * Supervises kitchen staff, manages inventory, and ensures compliance with health and safety regulations. |  |
| Delivery driver | * Delivers food orders to customers' homes or offices. * Ensures timely and accurate delivery, collects payments, and provides excellent customer service. |  |
| Dish washer | * Cleans and sanitizes dishes, utensils, and kitchen equipment. * Assists with maintaining cleanliness and organization in the kitchen. |  |
| Head waiter | * Takes orders from customers, serves food and beverages, and provides excellent customer service. * Answers questions about menu items, takes payments, and ensures a positive dining experience. |  |
| Barista | * mixes drinks, and serves beverages according to established recipes and customer preferences. * Maintains cleanliness and organization of the bar area, including stocking supplies and cleaning equipment. |  |
| Head Bar |  |  |
| Company | Human Resource Manager | * Manages recruitment, training, and development of employees across the company. * Implements HR policies and procedures, including compensation, benefits, and performance management. |  |
| Operation Manager | * Oversees overall operations of the company, including restaurant branches, production facilities, and distribution centers. * Develops and implements operational policies, procedures, and performance metrics. * Ensures compliance with health and safety regulations and company standards. |  |
| Logistics Coordinator | * Coordinates transportation and logistics activities, including scheduling deliveries and managing routes. * Communicates with suppliers, carriers, and internal teams to ensure timely delivery of goods. * Tracks shipments, monitors delivery performance, and resolves any transportation issues. |  |
|  |  |  |
|  |  |  |
|  |  |  |  |

**Accessibility Permissions**

|  |  |  |  |
| --- | --- | --- | --- |
| **Database User** | **Functions** | **View Name** | **View permission** |
| **Customer** |  |  |  |
| Branch Manager |  |  |  |
| Assistance Manager |  |  |  |
| Kitchen manager |  |  |  |
| Cashier |  |  |  |
| Chief |  |  |  |
| Delivery driver |  |  |  |
| Dish washer |  |  |  |
| Head waiter |  |  |  |
| Barista |  |  |  |
| Head Bar |  |  |  |
| Human Resource Manager |  |  |  |
| Operation Manager |  |  |  |
| Logistics Coordinator |  |  |  |
|  |  |  |  |
|  |  |  |  |