### **Normalization in DBMS**

Normalization is the process of organizing data to minimize redundancy and improve data integrity. The main goal is to ensure that the database is efficient and eliminates anomalies (update, insert, and delete anomalies). This is done by dividing large tables into smaller, related tables, and defining relationships between them.

**Normalization**

Let's take an example of an Inventory Management System for a retail store, where we store information about products, suppliers, and sales in an unnormalized table. This table might have redundant data and does not follow any normal form.

**Inventory Management System** normalization process with a table that initially does not follow any normal form (unnormalized), and we will apply **1NF, 2NF, 3NF, BCNF, 4NF, and 5NF** to gradually improve the structure of the database.

### **Unnormalized Table (Does not follow 1NF, 2NF, 3NF, BCNF, 4NF, or 5NF)**

Consider the following table

****

### **Key Observations:**

* Multiple supplier phone numbers in a single cell (violates atomicity of 1NF).
* Composite attributes like ItemsOrdered and SupplierPhones.
* Redundant information (e.g., repeated supplier details).

### 

### **First Normal Form (1NF)**

**Rule of 1NF**: Eliminate repeating groups and ensure each field contains atomic values. In 1NF, each cell must hold only one value, and there should be no multi-valued attributes.

To apply 1NF, we:

* Split multiple phone numbers into individual rows.
* Ensure ItemsOrdered contains atomic values (split rows for each ordered item).

#### 

#### **1NF Table:**

#### 

#### **Key Observations:**

* The table now conforms to 1NF, but we still have redundancy, such as repeated supplier information.

### 

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

**Rule of 2NF**: Eliminate partial dependency. A table is in 2NF if it is in 1NF and all non-key attributes are fully dependent on the entire primary key. This means that each non-key attribute must depend on the whole primary key (composite keys).

In our case:

* Attributes like SupplierName and SupplierPhone are only dependent on SupplierID, not on OrderID or ProductID. This violates 2NF.
* We need to split the table to remove partial dependencies.

#### 

#### **2NF Tables:**

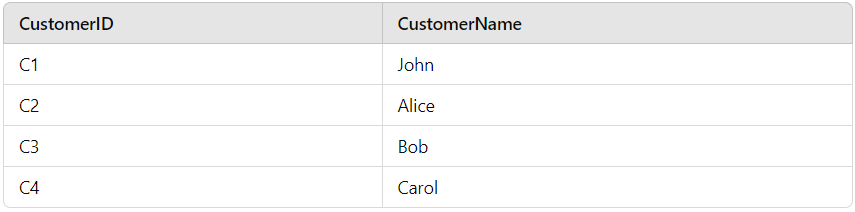
##### **Products Table:**

##### **Suppliers Table:**

##### 

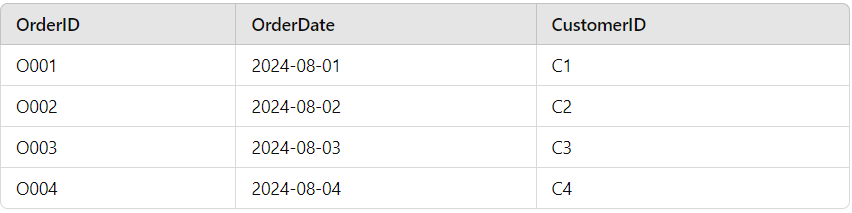
##### 

**Customers Table:** This new table links CustomerID to CustomerName, eliminating partial dependency in the Orders Table.

****

##### **Orders Table:**

The **CustomerName** is replaced by **CustomerID** to eliminate the partial dependency of **CustomerName** on **OrderID**, bringing the table into 2NF.



##### **OrderDetails Table:**

##### **Key Observations:**

* The table now follows 2NF, but there is still redundancy in supplier phone numbers.
* **Customer Table** ensures that customer information is managed separately, reducing redundancy and preventing partial dependency.
* **Orders Table** now references the **CustomerID** instead of **CustomerName**, ensuring that each non-key attribute is fully dependent on the primary key, satisfying 2NF.

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

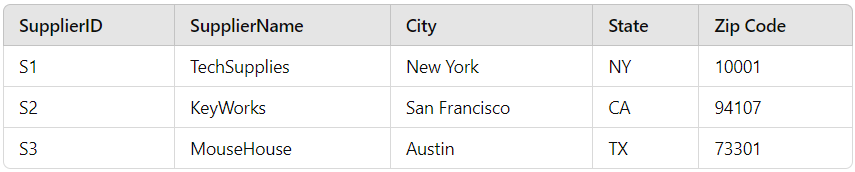
**Rule of 3NF**: Eliminate transitive dependencies. A table is in 3NF if it is in 2NF and all non-key attributes are non-transitively dependent on the primary key.

In the Suppliers table, the phone numbers still have a transitive dependency (multiple phone numbers for a single supplier). We can split it further.

#### **3NF Tables:**

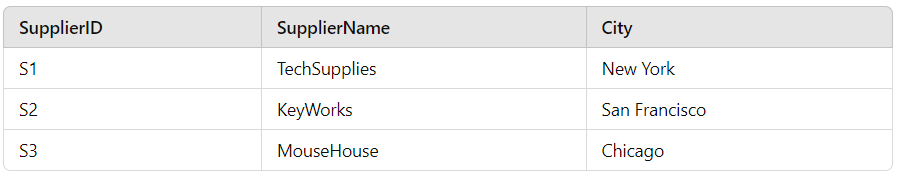
##### **Suppliers Table:**

Each supplier’s **SupplierID** will now be linked to **SupplierName**, **City**, **State**, and **Zip Code**. There is no transitive dependency in this table as all attributes are directly dependent on the primary key, **SupplierID**.

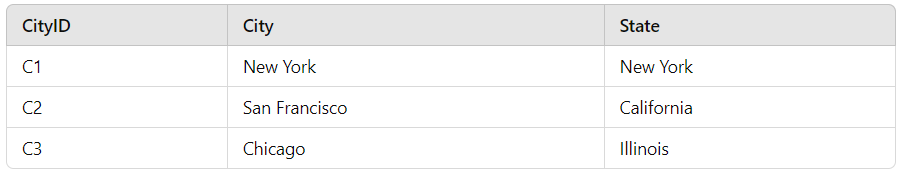


**Revised 3NF Tables:**

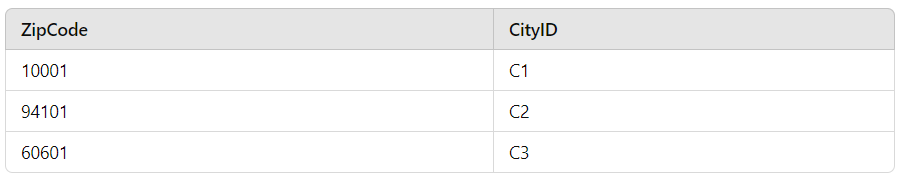
1. **Suppliers Table**

****

1. **Cities Table**

****

1. **ZipCodes Table**

****

##### **SupplierPhones Table:**

In this table, each **SupplierID** can have multiple phone numbers, but we’ve now separated phone numbers from the **Suppliers Table** to avoid transitive dependencies.

##### 

##### **Key Observations:**

* Now, all tables follow 3NF, and transitive dependencies have been eliminated.
* The **Suppliers Table** now only holds information directly related to the supplier (name, city, state, zip code) without redundant or transitive dependencies.
* **SupplierPhones** table separates the phone numbers, and each supplier can have multiple phone numbers, but we have yet to address the **multivalued dependency**, which we will tackle in **4NF**.

### 

### **Boyce-Codd Normal Form (BCNF)**

**Rule of BCNF**: A stronger version of 3NF, BCNF ensures that for every functional dependency, the left side is a superkey.

In this case, our tables already follow BCNF, as no non-trivial functional dependency violates the BCNF rule. The keys are properly structured.

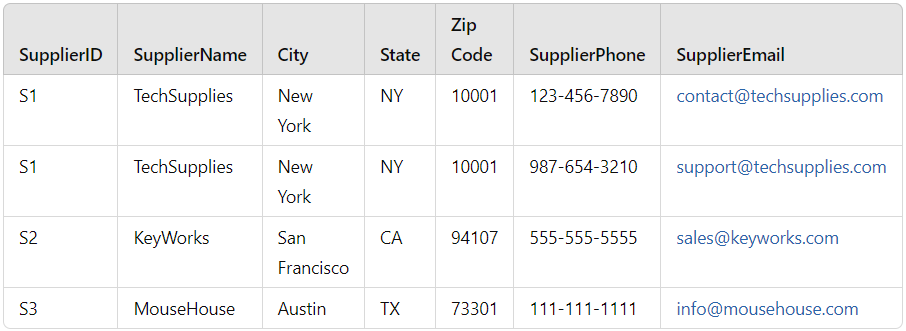
### **Fourth Normal Form (4NF)**

**Rule of 4NF**: A table is in **4NF** if it is free from **multivalued dependencies**, meaning a table should not contain two or more independent multivalued attributes for a single primary key. In this case, we have two multivalued attributes for each supplier: **SupplierPhone** and **SupplierEmail**.

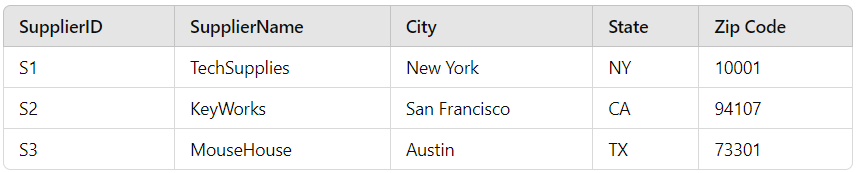
### **Unnormalized Table (Before 4NF):**

Let's say we started with a table that includes both **Phone Numbers** and **Emails** for each **Supplier**.

This structure has **multivalued dependencies**: each **Supplier** can have multiple **Phone Numbers** and **Emails**, which need to be split into separate tables to satisfy **4NF**.

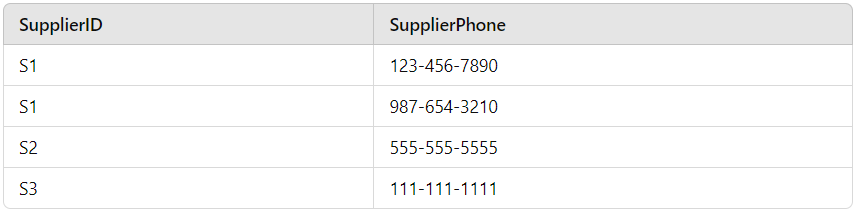


**Suppliers Table:**

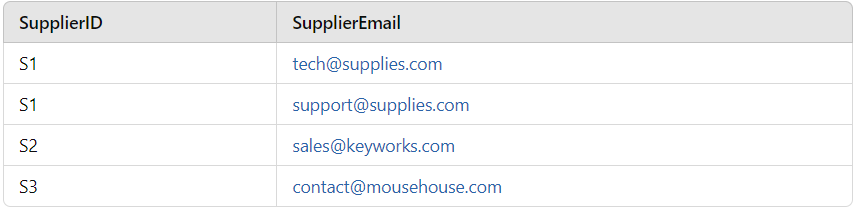
****

**SupplierPhones Table:**

This table stores phone numbers independently from other supplier attributes, avoiding any multivalued dependency



### **SupplierEmails Table**



### **Key Observations:**

* The original unnormalized table had **multivalued dependencies** between **SupplierPhones** and **SupplierEmails** for each **Supplier**.
* After decomposition, the **Suppliers Table** holds the main supplier details, while the **SupplierPhones** and **SupplierEmails** tables hold independent lists of phone numbers and emails, satisfying **4NF**.

### **Fifth Normal Form (5NF)**

**Rule of 5NF**: Eliminate join dependencies. A table is in 5NF if every join dependency is implied by the candidate keys.

Since our current structure does not have any complex join dependencies and no redundancy in decomposition, the tables already satisfy 5NF.