SQL Notes

September 23, 2025

Canine-Table

Github

POSIX Nexus serves as a comprehensive cross-language reference hub that explores the implementation and behavior of POSIX-compliant functionality across a diverse set of programming environments. Built atop the foundational IEEE Portable Operating System Interface (POSIX) standards, this project emphasizes compatibility, portability, and interoperability between operating systems.

Abstract

Contents

I Relationships and Keys **II**
I Relation II
I Cardinality VI
I Three Types of Minimum Cardinality VI

II Modelling **VII**

I Relationships and Keys

Alternative Terminology

Table	Column	Row
Relation	Attribute	Tuple
File	Field	Record

Equivalent Terminologies in Data Management Systems

I Relation

Characteristics of Relations

- Rows contain data about an entity.
- Columns contain data about attributes of the entities.
- All entries in a column are of the same kind.
- Each column has a unique name.
- Cells of the table hold a single value.
- The order of the columns is unimportant.
- The order of the rows is unimportant.
- No two rows may be identical.

create the example database

```sql
DELIMITER $$

CREATE PROCEDURE SwitchDatabase(IN dbName VARCHAR(64))
BEGIN
    -- Build the USE statement dynamically
    SET @sql = CONCAT('USE ', dbName);
    PREPARE stmt FROM @sql;
    EXECUTE stmt;
    DEALLOCATE PREPARE stmt;
END$$

DELIMITER ;

CREATE OR REPLACE DATABASE nexusDB
    CHARACTER SET utf8mb4
    COLLATE uca1400_as_cs
    COMMENT 'Where collations go to argue about accents and case,
↪but everything is encrypted anyway.';

CREATE DATABASE IF NOT EXISTS posixDB
    CHARACTER SET utf8mb4
    COLLATE uca1400_ai_ci
    COMMENT 'Because even databases deserve a POSIX-compliant
↪bedtime story.';

    SHOW DATABASES;
```

EXAMPLE OF AN EMPLOYEE RELATION

EmployeeNumber	FirstName	LastName	Department	EmailAddress	Phone
100	Jerry	Johnson	Accounting	JJ@somewhere.com	518-834-1101
200	Mary	Abernathy	Finance	MA@somewhere.com	518-834-2101
300	Liz	Smathers	Finance	LS@somewhere.com	518-834-3102
400	Tom	Caruthers	Accounting	TC@somewhere.com	518-834-1102
500	Ken	Jackson	Production	KJ@somewhere.com	518-834-2102
600	Eleanor	Caldera	Legal	EC@somewhere.com	518-834-3101
700	Richard	Bandalone	Legal	RB@somewhere.com	518-834-3102

mariadb code example

```sql
USE nexusDB
-- Create the Employee table
CREATE TABLE IF NOT EXISTS Employees (
        EmployeeNumber INT PRIMARY KEY,
        FirstName               VARCHAR(50) NOT NULL,
        LastName               VARCHAR(50) NOT NULL,
        Department           VARCHAR(50) NOT NULL,
        EmailAddress      VARCHAR(100) UNIQUE NOT NULL,
        Phone                           VARCHAR(20)
);

-- Insert employee records
```

```sql
INSERT INTO Employees (EmployeeNumber, FirstName, LastName, Department, EmailAddress, Phone) VALUES
(100, 'Jerry',     'Johnson',        'Accounting', 'JJ@somewhere.com', '518-834-1101'),
(200, 'Mary',         'Abernathy', 'Finance',         'MA@somewhere.com', '518-834-2101'),
(300, 'Liz', 'Smathers',     'Finance',         'LS@somewhere.com', '518-834-3102'),
(400, 'Tom',          'Caruthers', 'Accounting', 'TC@somewhere.com', '518-834-1102'),
(500, 'Ken',          'Jackson',         'Production', 'KJ@somewhere.com', '518-834-2102'),
(600, 'Eleanor', 'Caldera',        'Legal',             'EC@somewhere.com', '518-834-3101'),
(700, 'Richard', 'Bandalone', 'Legal',             'RB@somewhere.com', '518-834-3102');
```

Employee Directory with Multiple Phone Entries

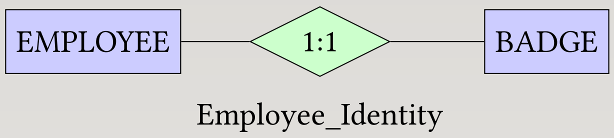
EmployeeNumber	FirstName	LastName	Department	EmailAddress	Phone
100	Jerry	Johnson	Accounting	JJ@somewhere.com	518-834-1101
200	Mary	Abernathy	Finance	MA@somewhere.com	518-834-2101
300	Liz	Smathers	Finance	LS@somewhere.com	518-834-2102
400	Tom	Caruthers	Accounting	TC@somewhere.com	Fax: 518-834-9711 Home: 518-834-9915
500	Tom	Jackson	Production	TJ@somewhere.com	518-834-3101
600	Eleanore	Caldera	Legal	EC@somewhere.com	Fax: 518-834-9711 Home: 518-834-9915
700	Richard	Bandalone	Legal	RB@somewhere.com	518-834-3102

mariadb code example

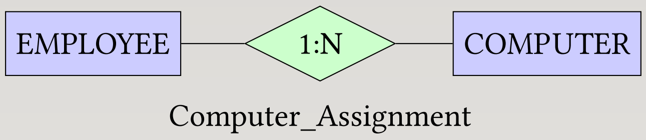
```
USE nexusDB
    -- Create Employees table if it does not exist
    CREATE TABLE IF NOT EXISTS Employees (
        EmployeeNumber INT PRIMARY KEY,
        FirstName              VARCHAR(50) NOT NULL,
        LastName               VARCHAR(50) NOT NULL,
        Department          VARCHAR(50) NOT NULL,
        EmailAddress     VARCHAR(100) UNIQUE NOT NULL
    );

    -- Create EmployeePhones table if it does not exist
    CREATE TABLE IF NOT EXISTS EmployeePhones (
        PhoneID                    INT AUTO_INCREMENT PRIMARY KEY,
        EmployeeNumber INT NOT NULL,
        PhoneType              VARCHAR(20) NOT NULL,      -- e.g.
↪'Work', 'Fax', 'Home'
        PhoneNumber          VARCHAR(30) NOT NULL,
        FOREIGN KEY (EmployeeNumber) REFERENCES
↪Employees(EmployeeNumber)
    );

    DELIMETER$$
    CREATE PROCEDURE AddEmployeeWithPhone(
        IN pEmployeeNumber INT,
        IN pFirstName              VARCHAR(50),
        IN pLastName               VARCHAR(50),
        IN pDepartment          VARCHAR(50),
        IN pEmailAddress     VARCHAR(100),
        IN pPhoneType              VARCHAR(20),
        IN pPhoneNumber          VARCHAR(30)
    ) BEGIN
    -- If employee does not exist, insert them
    IF NOT EXISTS (
        SELECT 1 FROM Employees WHERE EmployeeNumber =
↪pEmployeeNumber
    ) THEN
        INSERT INTO Employees (EmployeeNumber, FirstName,
↪LastName, Department, EmailAddress)
        VALUES (pEmployeeNumber, pFirstName, pLastName,
↪pDepartment, pEmailAddress);
    END IF;

        -- Insert the phone entry (can be multiple per employee)
        INSERT INTO EmployeePhones (EmployeeNumber, PhoneType,
↪PhoneNumber)
        VALUES (pEmployeeNumber, pPhoneType, pPhoneNumber);
    EN$$

    DELIMITER ;

    CALL AddEmployeeWithPhone(
            800, 'Alice', 'Walker', 'HR', 'AW@somewhere.com',
                'Work', '518-834-4101'
        );
```

I Cardinality

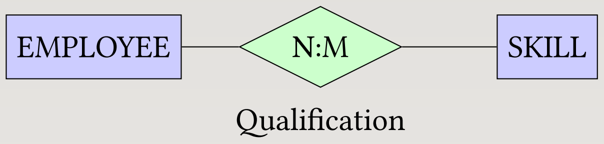
I Three Types of Minimum Cardinality



(a) Mandatory-to-Mandatory (M-M) Relaitonship



(b) Optional-to-Optional (O-O) Relaitonship

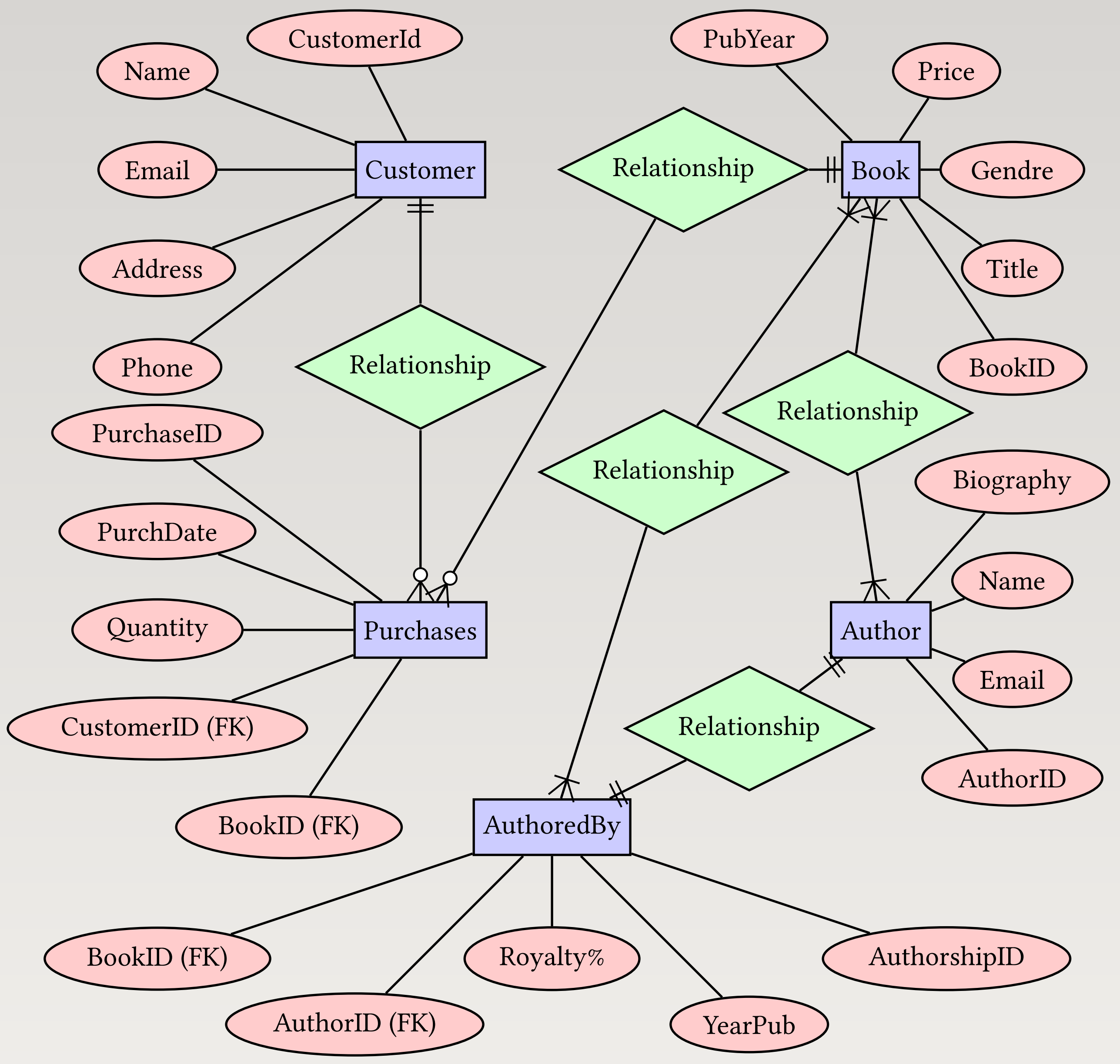


(c) Optional-to-Mandatory (O-M) Relaitonship

II Modelling

Scenario 0 (Book Store System)

- A book store is designing a database system to manage its inventory, customers, authors, and purchase records. The following describes the structure of their data.
- Customers are tracked by their name, email, address, phone number, and the date of their purchases.
- Each customer has a unique CustomerID.
- Books are tracked by their title, genre, price, and publication year.
- Each book has a unique BookID.
- Authors are tracked by their name, email, and biography.
- Each author has a unique AuthorID.
- Customers can purchase multiple books, and each purchase records the quantity and date.
- Each purchase links a CustomerID and a BookID, forming a many-to-many relationship between customers and books.
- Books may be authored by multiple authors, and authors may write multiple books.
- The AuthoredBy relationship tracks which author wrote which book, along with the royalty percentage, year of publication, and a unique AuthorshipID.
- Books and authors are also directly linked through a many-to-many relationship, separate from AuthoredBy.
- The system uses relationship nodes to clarify cardinalities, such as mandatory or optional participation and one-to-many or many-to-many connections.


CustomerId
Name
PubYear
Price
Email
Customer
Relationship
Book
Gendre
Address
Title
Phone
Relationship
BookID
PurchaseID
Relationship
Relationship
Biography
PurchDate
Name
Quantity
Purchases
Author
Email
CustomerID (FK)
Relationship
AuthorID
BookID (FK)
AuthoredBy
BookID (FK)
Royalty%
AuthorshipID
AuthorID (FK)
YearPub

Scenario 1 (Veterinary Hospital)

- A local veterinary hospital is looking for a replacement for their patient tracking system. When asked to describe their data, they responded as follows.
- Clients may have one or more pets.
- Clients are identified by their phone number.
- Clients are tracked by their name and address.
- A pet is owned by a single client only.
- A pet is identified by their owner's name and the pet name.
- Pets are tracked by their species, breed, sex, and neutering status.
- A pet can be treated by several different doctors and/or technicians (staff).
- Each doctor and technician has an employee number and a name.
- A pet may have multiple visits that are tracked by date and reason for visit.

Scenario 2 (Software Company)

- A software development company is modeling its internal structure and project assignments.
- The company has many employees.
- Employees are tracked by their name and a unique employee ID.
- The company has several departments, such as engineering, quality assurance, and tech support.
- Each department has a name and a manager.
- Managers are also employees and have unique employee IDs.
- Each department must have at least one employee assigned to it.
- Employees must be assigned to at least one department, but may belong to multiple departments.
- Projects are tracked by a unique project ID and a project name.
- Each project must have at least one employee assigned to it.
- An employee may be assigned to zero or more projects.
- Departments, projects, managers, and employees are all tracked by their names.