SQL Databases

Day #5

Wholeness

- In this lesson, we will learn the concepts, principles and techniques for how to create suitable Database design and how to implement it for an enterprise application software solution.
- A Database is an organized, structured collection of data, useful to an enterprise.
- Science of Consciousness: *Order is present everywhere.*

1.1 Overview

Meaning of essentials terms:

- Data: is facts or figures collected together through observation or measurement, for the purpose of referencing or analysis.
- Information: is what is derived, when data has been made to become meaningful or useful through processing, interpreting or presenting.
- Database: is a structured collection of related data.
- Database Management system (DBMS): is software that manages and controls access to and operation of, a database.

1.1 Overview

- Database Application: is any computer program that interacts with a database at some point during its execution.
- Database system: is the collection of application programs that interact with the database, together with the DBMS and the database itself.

Note: All access to the Database is through the DBMS.

1.1 Overview

Why study Databases?

- The Database system is perhaps the most important development in the field of software engineering.
- A Database typically forms the underlying framework of the information system on which most organizations operate.
- The importance of database systems keeps increasing, driven by significant developments in hardware capability/capacity and data communication and the emergence of the Internet, eCommerce, Business Intelligence and network/grid computing.

1.1 Overview

- Some areas of application of Database systems:
  - Purchases from the supermarket.
  - Online purchases using your credit card.
  - Booking a vacation with a travel agency
  - Using the local/school library
  - Taking out insurance
  - Finding and watch a movie
  - Conducting online or traditional banking
  - Studying at a college or university
  - Using the Internet/World-wide websites

Main Point 1

- It is important that we understand what Databases are, their usefulness within the context of an overall enterprise information system and the related terminologies surrounding the study of Database systems. *Knowledge is different in different states of consciousness.*

Relational Database

Non-Relational Database

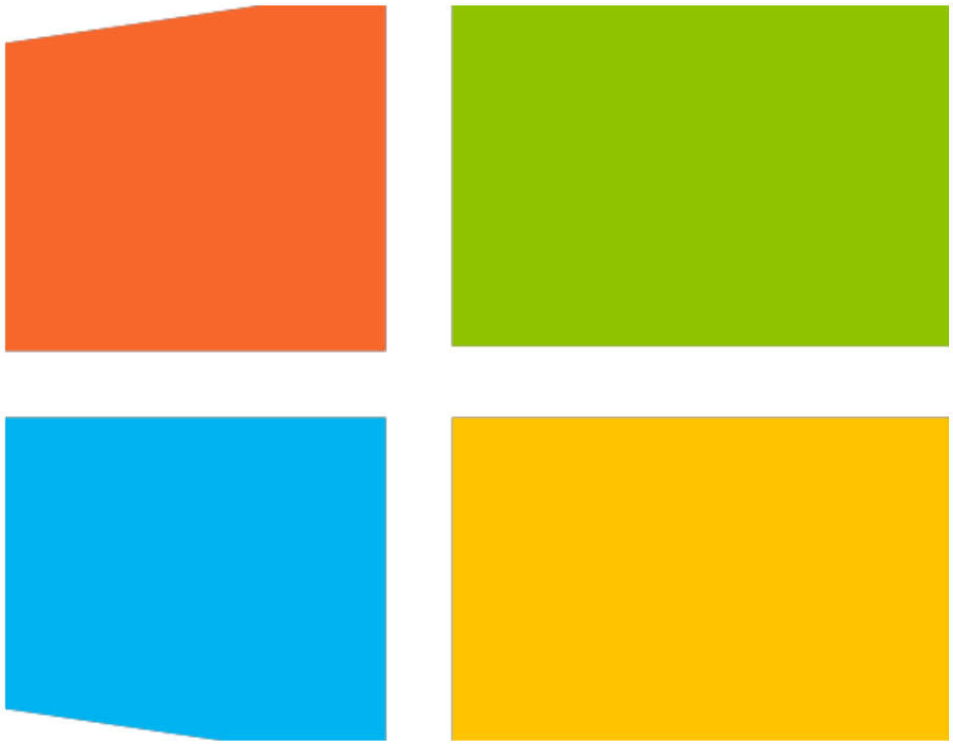
1.10 Introduction to MySQL

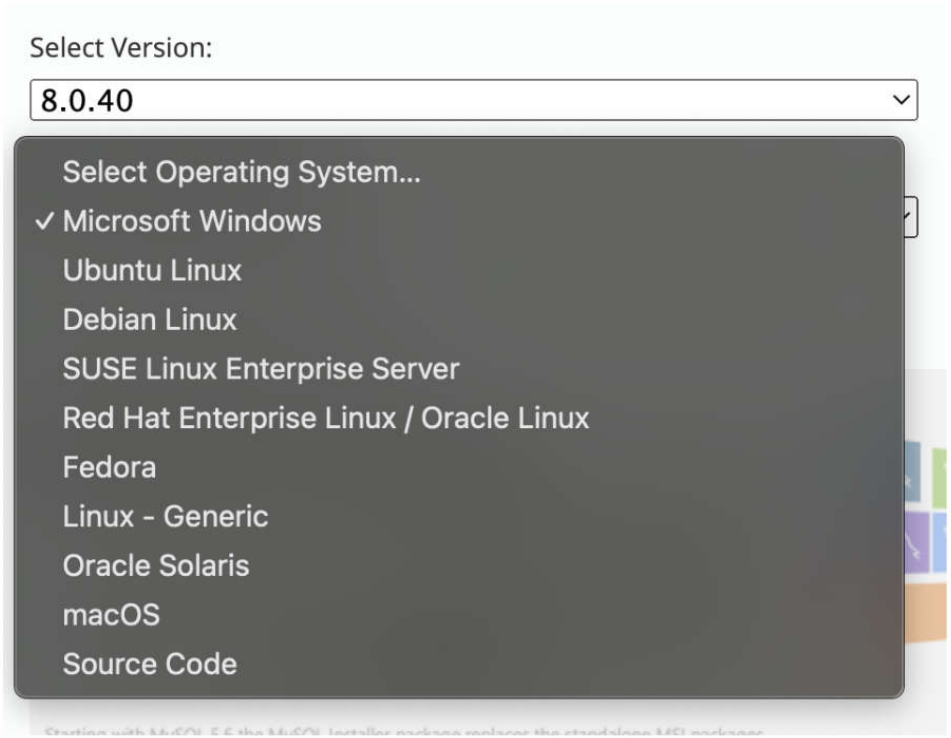
- MySQL is an open-source relational database management system (RDBMS)
- In this course we will be studying and implementing the various Database Management system concepts and techniques using a MySQL instance
- Therefore, you are required to download and install a MySQL Database server on your local machine (if you do not already have it).*
- MySQL can be obtained from https://dev.mysql.com/

***Note: A demonstration of how to obtain and install MySQL will be given.**

Windows

https://dev.mysql.com/downloads/installer/





MySQL

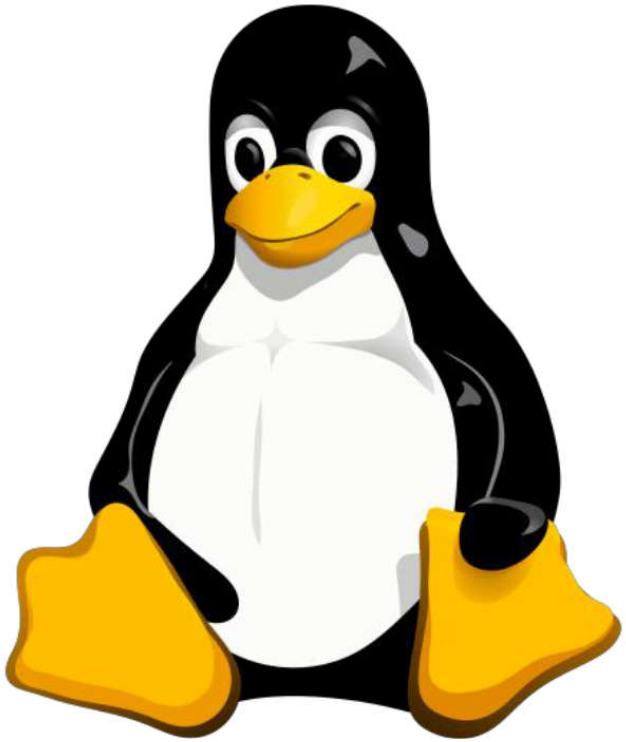
Update your package index
sudo apt update

Install MySQL server
sudo apt install mysql-server

Start MySQL service
sudo systemctl start mysql

Enable MySQL to start on boot
sudo systemctl enable mysql

Check the MySQL version to confirm the installation
mysql --version



Mac OS

brew install mysql

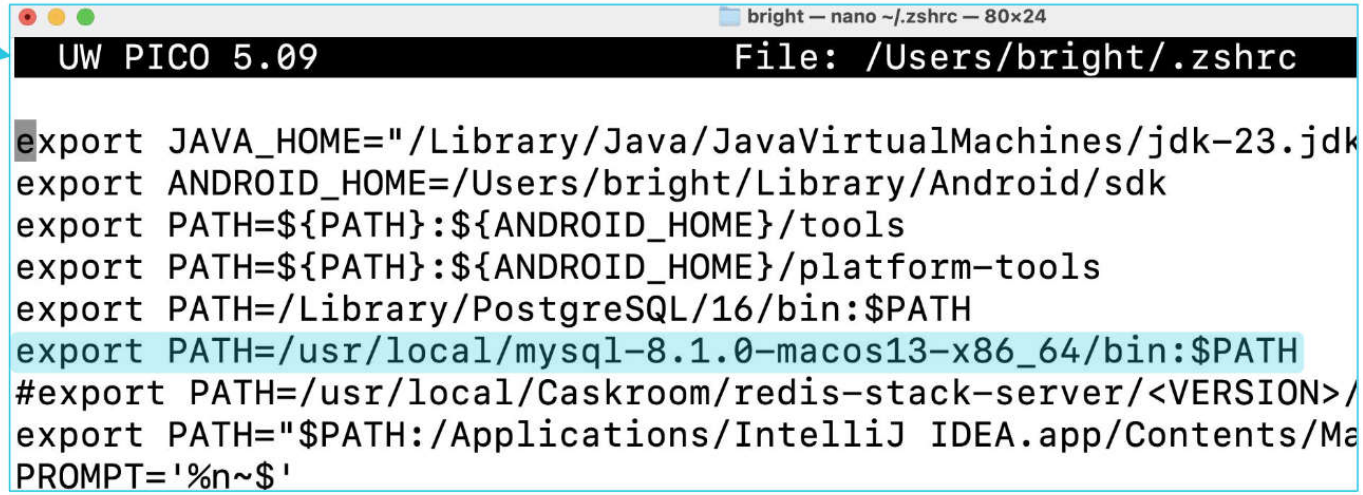
brew services start mysql

MacOS

```
bright~$mysql --version
zsh: command not found: mysql
```

```
bright~$sudo find / -name mysql
Password:
/usr/local/mysql-8.1.0-macos13-x86_64/bin/mysql
/usr/local/mysql-8.1.0-macos13-x86_64/include/mysql
/usr/local/mysql-8.1.0-macos13-x86_64/data/mysql
```

```
bright~$nano ~/.zshrc
bright~$source ~/.zshrc
```



Mac OS

- bright~$mysql --version

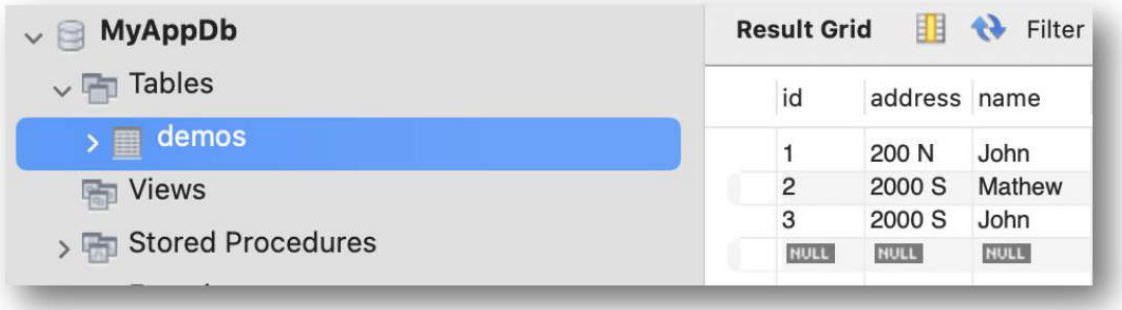
mysql Ver 8.1.0 for macos13 on x86_64 (MySQL Community Server - GPL)

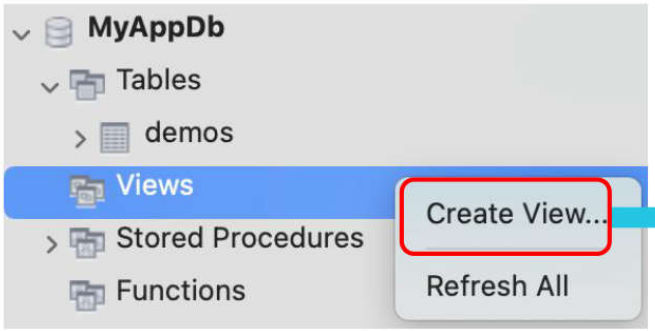
- bright~

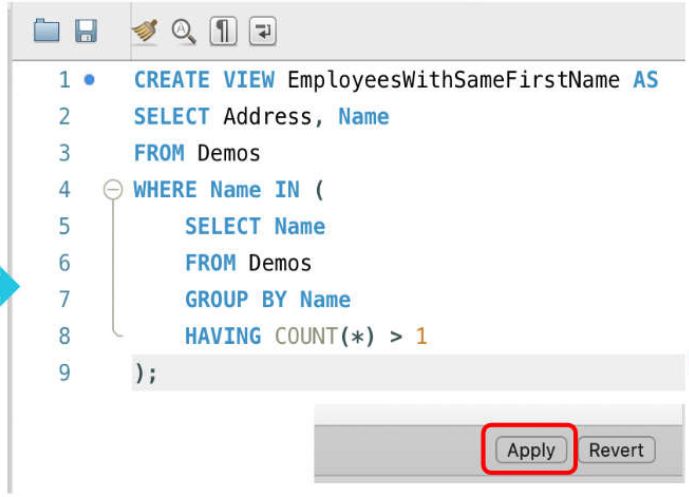
Data Modeling & Database Design

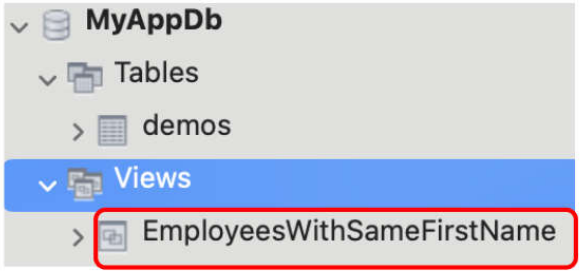
- Conceptual database model
  - Entity-Relationship model (E-R diagram)
  - Not tied to any specific RDBMS, yet
- Physical data model
  - Implementation of the E-R model on to a physical RDBMS using DB tables, views, stored procedures etc.
  - Includes RDBMS-specific data types and constructs

Create a view to filter employees with the same name using a subquery

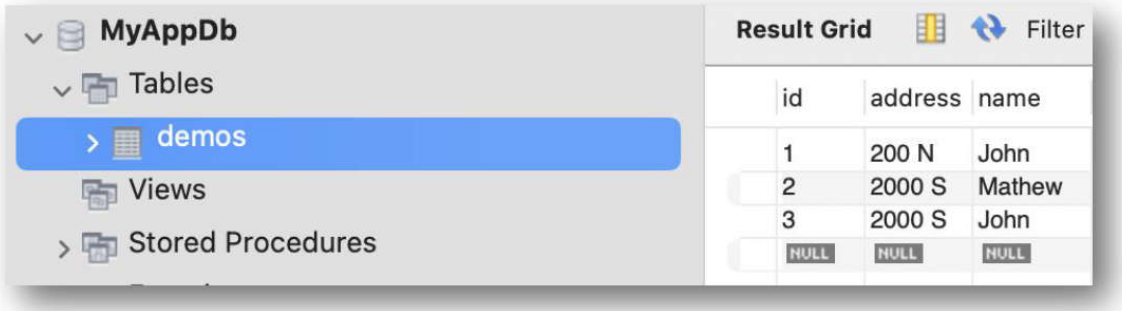






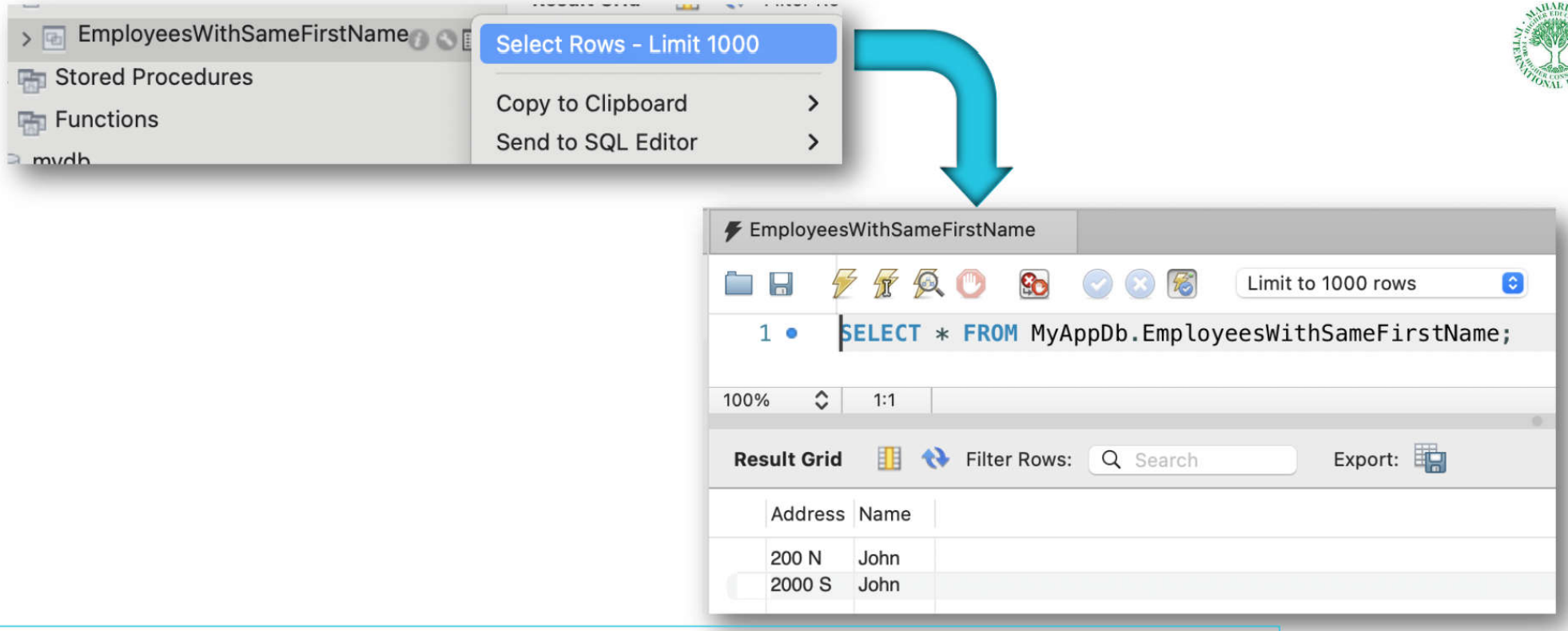


Create a view to filter employees with the same name using a subquery




```
CREATE VIEW EmployeesWithSameFirstName AS
SELECT Address, Name
FROM Demos
WHERE Name IN (
  SELECT Name
  FROM Demos
  GROUP BY Name
  HAVING COUNT(*) > 1
);
```

```
CREATE VIEW EmployeesWithSameFirstName2 AS
SELECT Address, Name
FROM Demos
WHERE Name IN (
  'John'
);
```

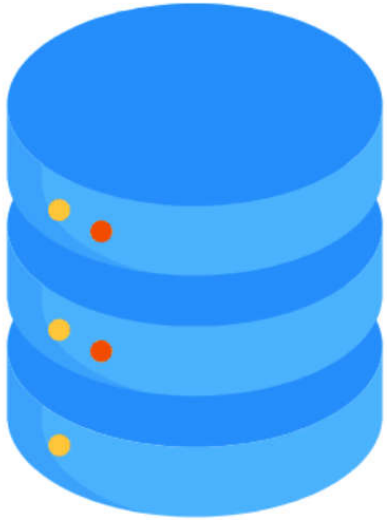



```
Statement stmt = con.createStatement();
ResultSet rs = stmt.executeQuery("SELECT * FROM EmployeesWithSameFirstName")){

while (rs.next()) {
   System.out.println(rs.getString("Address") + " " + rs.getString("Name"));
}
```

Implementing Relational Database

- Two possible approaches:
  - Bottom-up (a.k.a Database-First approach)
    - The Database objects (tables, views etc.) are created first, before the application code is written
    - This can be down either Interactively, using a Database client tool, such as MySQL Workbench or SQL Script
  - Top-down (a.k.a Code-First approach)
    - The Domain Entity classes are first coded in the application
    - And then an Object-Relational Mapping framework/tool is used to generate the corresponding Database objects.



The database design process aims to create database structures that will efficiently store and manage data (Rob & Coronel, 2004)
Ref: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=317584283d177f704b41c0256847888b1b95f47b

- A hybrid approach is most effective. This involves starting with a top-down design to establish a high-level overview of the system and then using a bottom-up approach to develop and test individual components.
- Ultimately, the best approach is the one that best suits the specific needs of the project.

Programming Language	ORM Tools	Description
Java	Hibernate	Mature and widely used, offering powerful and flexible object-relational mapping.
	EclipseLink	Open-source Java persistence platform supporting various data services.
	JPA	Standard specification for accessing relational databases from Java applications.
	DataNucleus	Open-source JDO and JPA implementation for persistence across various data stores.
	jOOQ	Fluent API for building type-safe SQL queries, alternative to string-based SQL.
Python	Django ORM	Integrated ORM in the Django web framework, offering high-level database access.
	SQLAlchemy	Powerful and versatile ORM, supporting various database backends and features like object relationships and unit of work patterns.
	SQLObject	Lightweight and easy-to-use ORM for Python, suitable for simpler projects.
C# and .NET	Entity Framework (EF)	Popular Microsoft-developed ORM, providing a powerful and mature solution for data access.
	Dapper	Lightweight micro-ORM for .NET, offering a simpler approach to database interactions.
	NHibernate	.NET port of the popular Hibernate ORM tool, providing similar features for C#.
Ruby on Rails	ActiveRecord	Powerful and well-integrated ORM included in the Rails framework for Ruby applications.
Node.js	TypeORM, Sequelize	Popular choices for relational database access in Node.js environment.

Database Design and Implementation Demo

- **Exercise**:
  - Create an E-R model for the City Library system. You may draw the ER diagram using a graphical drawing tool on computer or draw by hand.
  - Implement the model on a physical Relational database. Use any RDBMS of your choice e.g. MySQL

Symbols in Crow's Foot Notation and Their Meaning

Zero

One

Many

Three symbols are used to represent cardinality:

ER diagram

A ring represents "zero"

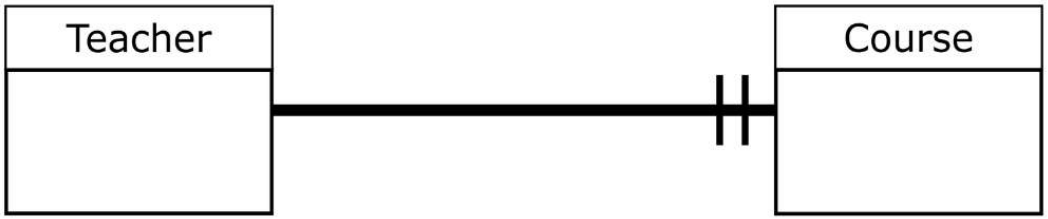
A dash represents "one"

A crow's foot represents "many" or "infinite"

Four types of Cardinality

Ring and dash: Minimum zero, maximum one (optional)	
Dash and dash: Minimum one, maximum one (mandatory)	
Ring and crow's foot: Minimum zero, maximum many (optional)	
Dash and crow's foot: Minimum one, maximum many (mandatory)	

One and Only One

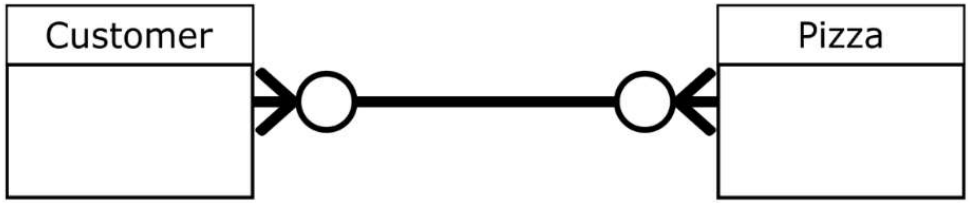


The minimum number of courses a teacher can take up is one, and the maximum is also one.

One or Many

- For each course, we want to have one or many teachers to choose from – meaning that one course can be taught by one or many teachers. The minimum here will be one while the maximum will be many.
- The notation to be used is **one or many**. The notation will be placed on the left side of the horizontal line.

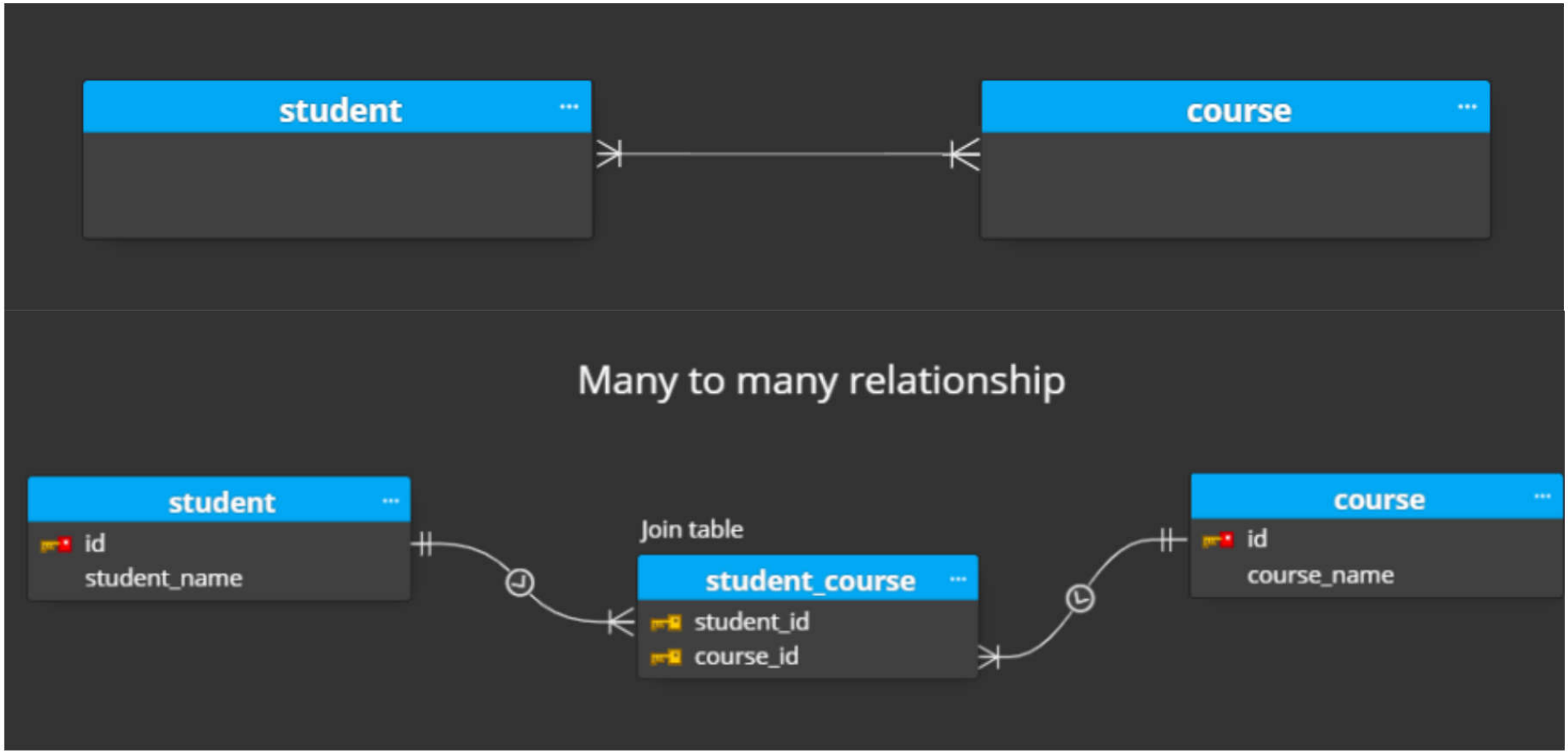
Zero or Many



- Zero is the minimum while many is the maximum.

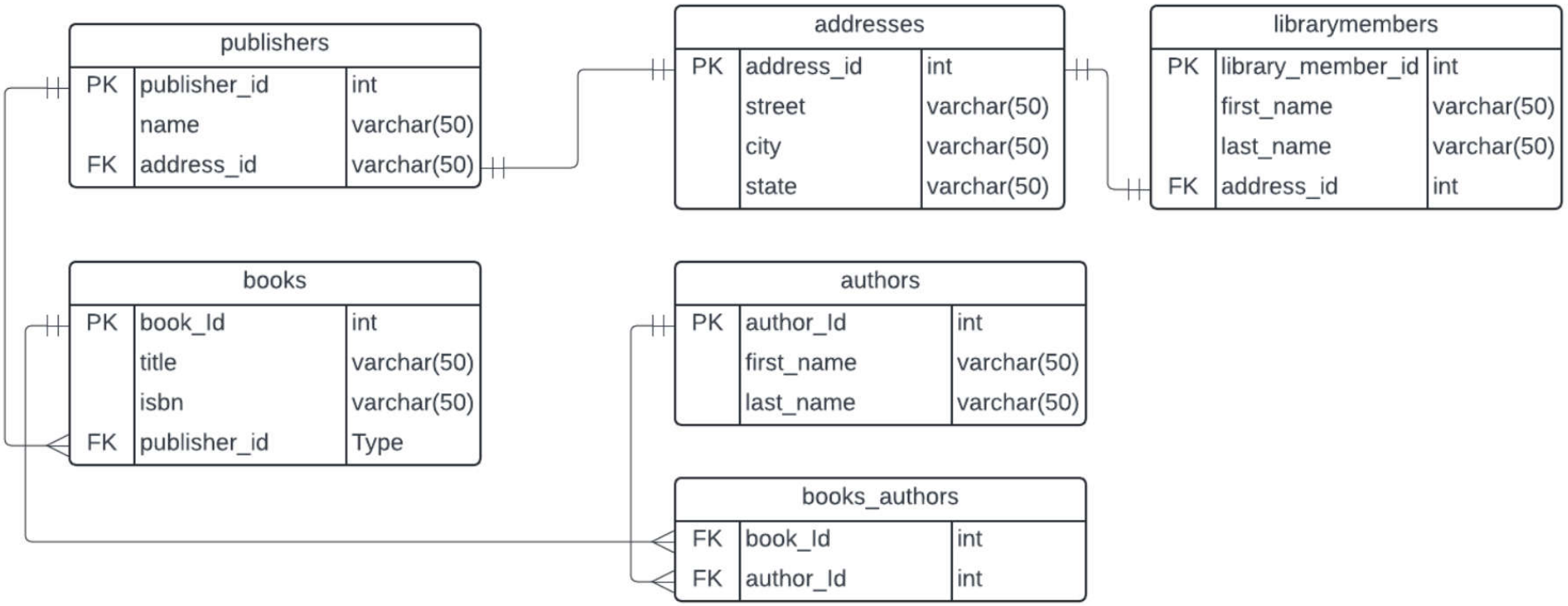
- Let's begin with the notation on the left. It has the **zero or many** notation. This implies that a pizza can be ordered by none (optional) or many customers.
- Similarly, the notation on the right side implies that a customer can order **zero or many** pizzas.

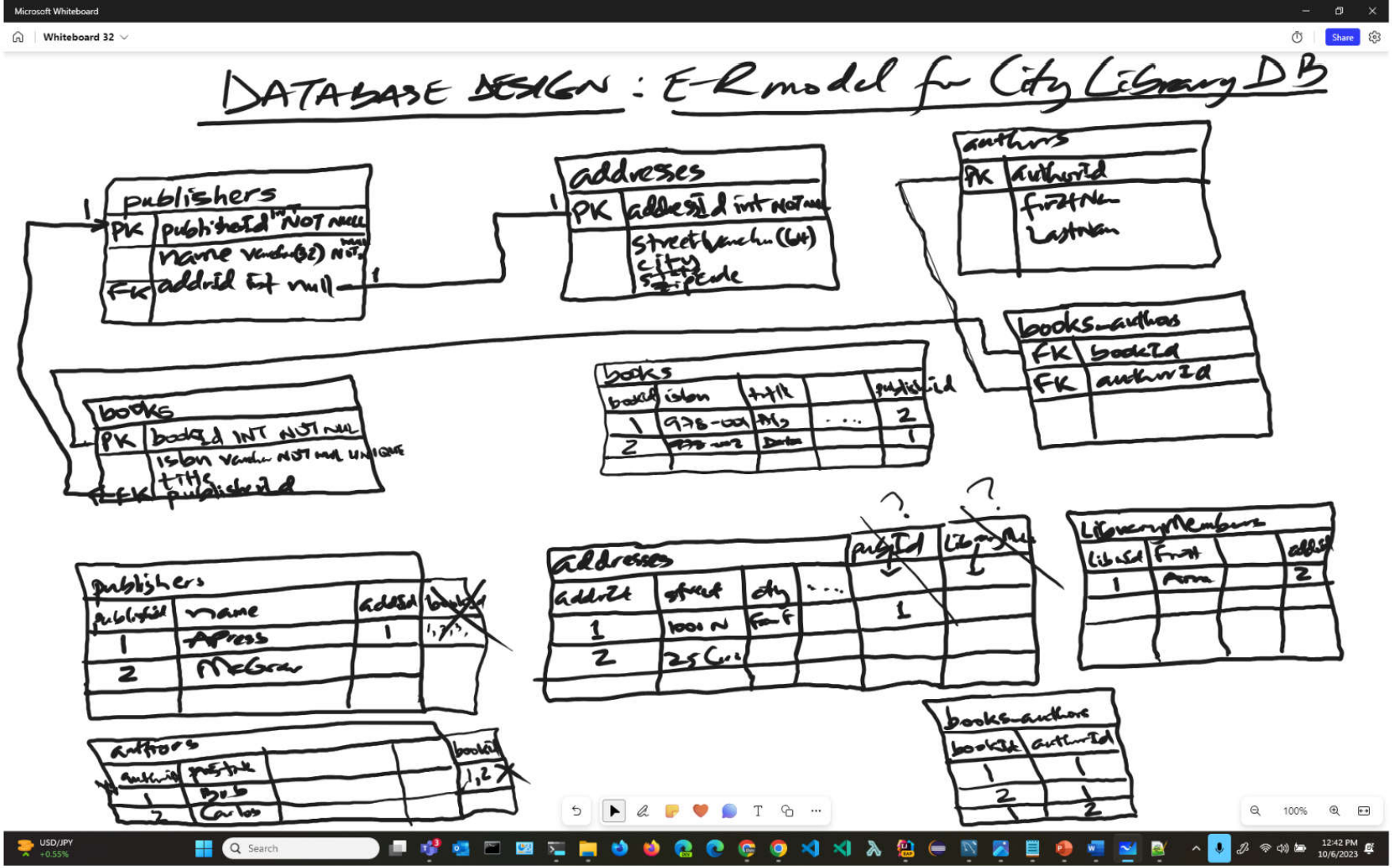
Many to Many



Ref: https://www.datensen.com/blog/er-diagram/many-to-many-relationships/

Partial Solution




Microsoft Whiteboard
Whiteboard 32
Share
DATABASE DESIGN : E-R model for City Library DB
publishers
addresses
authors
books_authors
books
books
addresses
authors
books_authors
LibraryMembers
100%
USD/JPY
Search
12:42 PM
10/6/2023

Validations

Annotation	Applies To	Condition
@NotNull	Any field type	Field cannot be null
@NotEmpty	String, Collection, Map, Array	Not null and size greater than zero
@NotBlank	String	Not null, not empty, and trimmed (no whitespace only)

JPA Query Method and JPQL

```
public interface PublisherRepository extends JpaRepository<Publisher, Integer> {
   @Query("select p from Publisher p")
   List<Publisher> findAllData();

   Optional<Publisher> findByPublisherName(String publisherName);

   @Query("SELECT p FROM Publisher p WHERE p.publisherName = :publisherName")
   Optional<Publisher> findDataUsingPublisherName(@Param("publisherName") String publisherName);
}
```

```
@Entity   @Table(name = "publ
public class Publisher {
   @Id
   @GeneratedValue(strategy = G
   @Column(name = "publisher_i
   private Integer publisherId;
   @Column(name = "publisher_n
   private String publisherName;
   @Embedded
   private Address address;

   public Publisher(String publishe
      this.publisherName = publish
   }
   //...
}
```

Added dependencies:

- Lombok
- JDBC API
- Spring Data JPA
- MySQL Driver
- Validation

@Column and @JoinColumn

@Column annotation:

•Used on fields within an entity class.

•Defines how a specific field in your entity class maps to a column in the database table.

@JoinColumn annotation:

•Used within a one-to-one or many-to-one relationships between entities.

•Specifies how a foreign key column in the owning entity table is mapped to the primary key of the referenced entity table.

Live Demo