



CICLO 2

[FORMACIÓN POR CICLOS]

Programación Básica JAVA

Semana 5



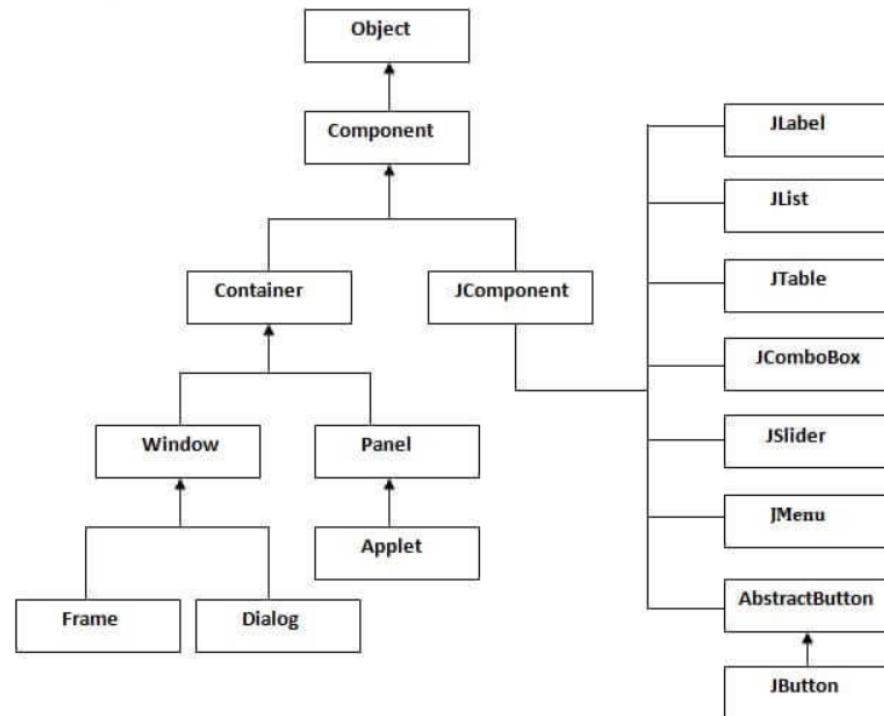
Programa

Semana 5	Introducción a las pruebas unitarias con Junit	4	10	14
	Introducción a la persistencia mediante bases de datos	3	7	10

Reto 2

Socializar Reto 2

Repaso de interfaces Java SWING



Revisión de interfaces Java SWING

<https://quizizz.com/join?gc=38240742>

Pruebas de Software

Las pruebas de software se definen como el proceso que ayuda a identificar la ***corrección, completitud, seguridad y calidad*** del software desarrollado. De forma genérica, por software se entiende el conjunto de datos, documentación, programas, procedimientos y reglas que componen un sistema informático

Pruebas de Software

Prueba es el proceso de ejecutar un conjunto de elementos software con el fin de encontrar errores. Por tanto, probar no es demostrar que no hay errores en el programa ni únicamente mostrar que el programa funciona correctamente.

Caso de prueba de software conjunto de condiciones, datos o variables bajo las cuales el desarrollador determinará si el o los correspondientes requisitos de un sistema software se cumplen de forma parcial, de forma completa o no se cumplen

Pruebas de Software

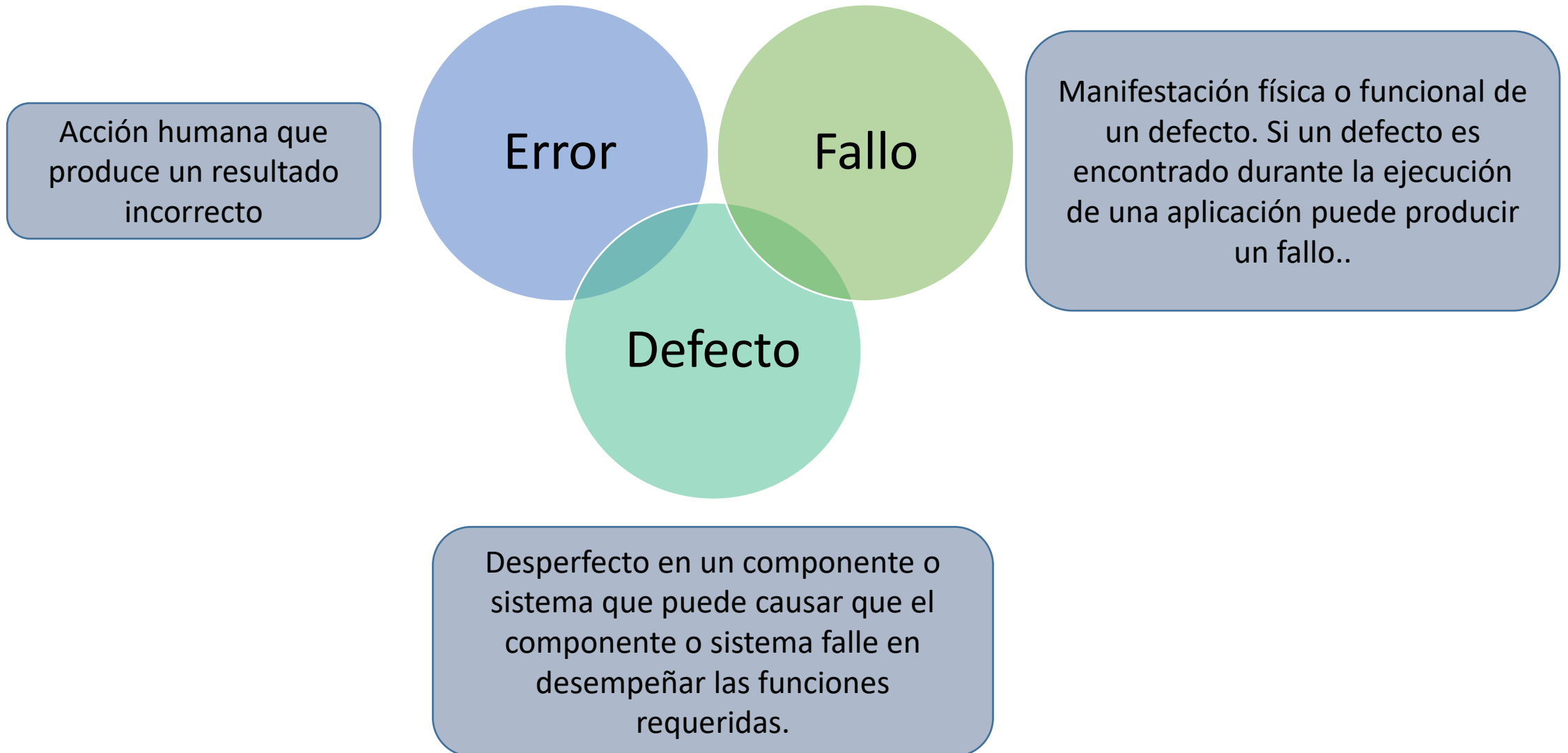
Otros conceptos fundamentales son los de **error, fallo y defecto software**.

- **Error** la discrepancia entre un valor o condición calculado, observado o medido y el valor o condición específica o teóricamente correcta. Es un fallo que comete el desarrollador.
- **Defecto técnico** es la desviación en el valor esperado para una cierta característica.
- **Fallo** es la consecuencia de un defecto.

En muchas ocasiones estos términos se confunden y se utiliza uno de ellos de forma genérica.

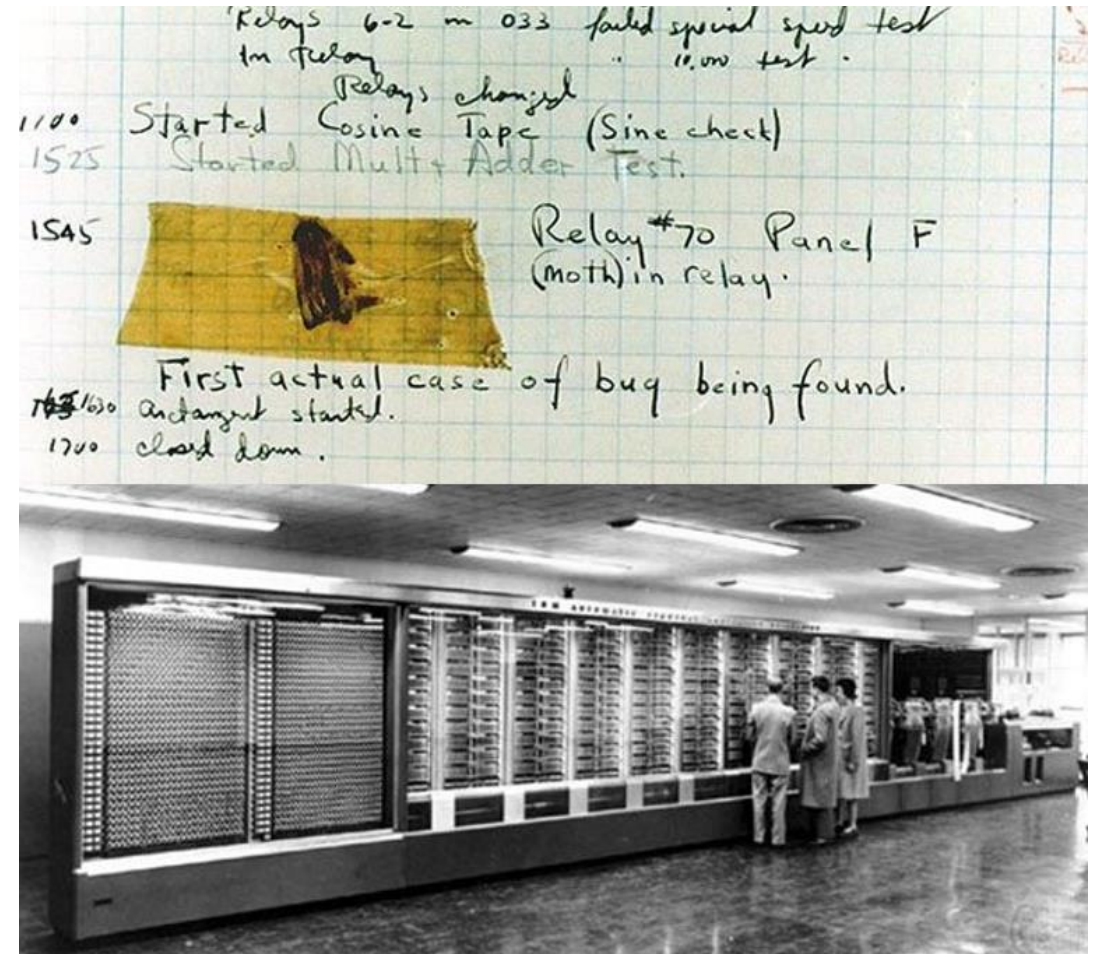
El objetivo de las pruebas es, por tanto, descubrir los errores y fallos cometidos durante las fases anteriores de desarrollo del producto.

Pruebas de Software



Pruebas de Software

El 9 de septiembre de 1947, la física y matemática Grace Murray Hopper y otros que trabajaban en la Universidad de Harvard en el Mark II informaron de que el ordenador sufrió un fallo en el relé electromagnético #70 del panel F. Cuando se investigó ese relé, el equipo encontró una polilla (bug) frita que provocó que el relé quedase abierto. Hopper pegó el insecto con cinta adhesiva en la bitácora con el comentario



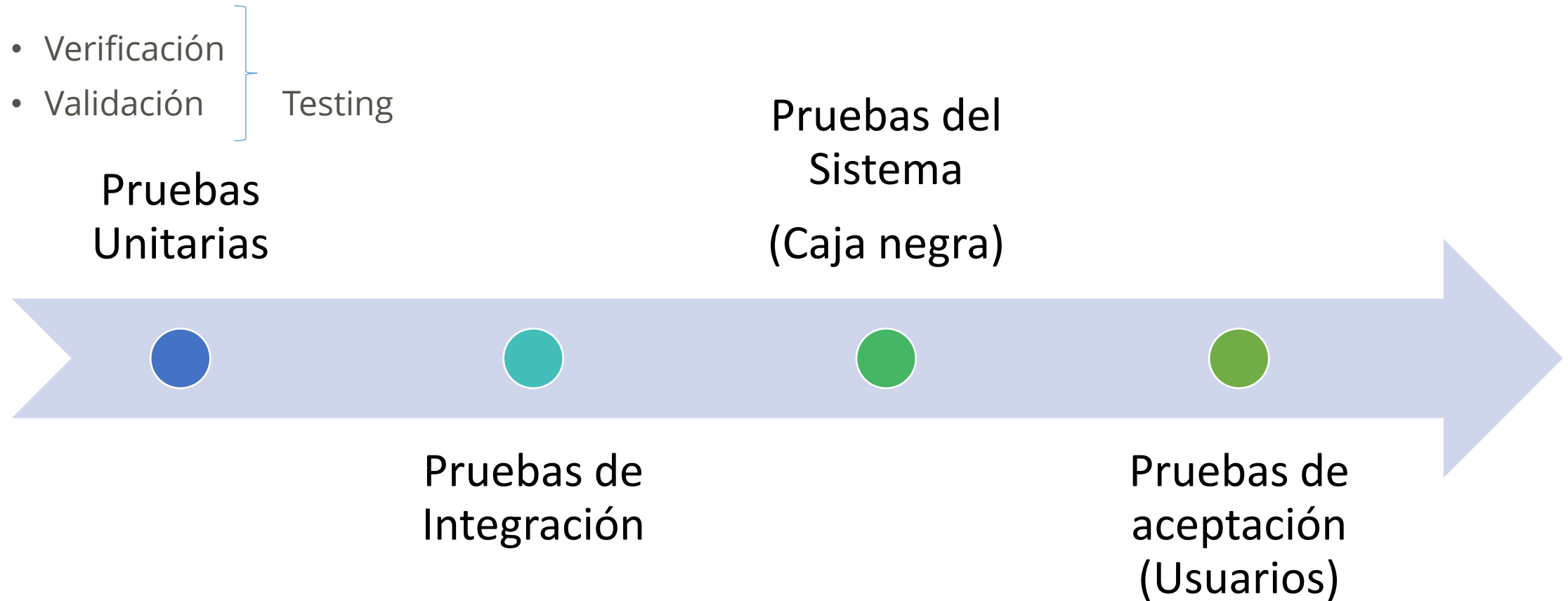
Pruebas de Software

Otros conceptos importantes son los de validación y verificación:

La verificación comprueba el funcionamiento del software, es decir, asegura que se implemente correctamente una funcionalidad específica. En definitiva, responde a la pregunta ***¿se ha construido el sistema correctamente?***

La validación comprueba si los requisitos de usuario se cumplen y los resultados obtenidos son los previstos. Responde a la pregunta ***¿se ha construido el sistema correcto?***

Pruebas unitarias - Conceptos



Pruebas unitarias - Conceptos

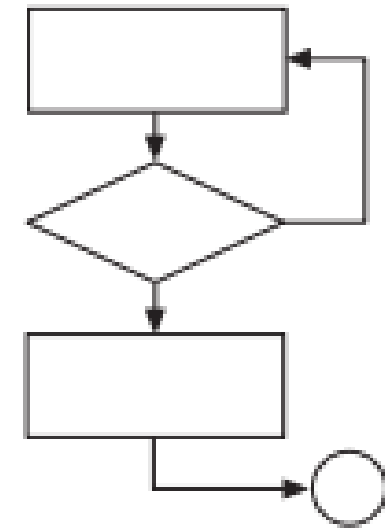
Pruebas Manuales	Pruebas Automatizadas
La ejecución manual de casos de prueba sin el soporte de ninguna herramienta se conoce como prueba manual.	Tomar el soporte de la herramienta y ejecutar los casos de prueba mediante el uso de una herramienta de automatización se conoce como automatización de pruebas.
Consume mucho tiempo y es tedioso : dado que los casos de prueba los ejecutan recursos humanos, es muy lento y tedioso.	Rápido : la automatización ejecuta casos de prueba significativamente más rápido que los recursos humanos.
Gran inversión en recursos humanos : dado que los casos de prueba deben ejecutarse manualmente, se requieren más probadores en las pruebas manuales.	Menos inversión en recursos humanos : los casos de prueba se ejecutan utilizando herramientas de automatización, por lo que se requiere menos cantidad de probadores en las pruebas de automatización.
Menos confiable : las pruebas manuales son menos confiables, ya que deben tener en cuenta los errores humanos.	Más confiable : las pruebas de automatización son precisas y confiables.
No programable : no se puede realizar ninguna programación para escribir pruebas sofisticadas para obtener información oculta.	Programable : los probadores pueden programar pruebas sofisticadas para revelar información oculta.

Pruebas de caja blanca

Estas pruebas se centran en probar el comportamiento interno y la estructura del programa examinando la lógica interna, como muestra la en la figura:

- Se ejecutan todas las sentencias (al menos una vez).
- Se recorren todos los caminos independientes de cada módulo.
- Se comprueban todas las decisiones lógicas.
- Se comprueban todos los bucles.

Finalmente, en todos los casos se intenta provocar situaciones extremas o límites



Pruebas Unitarias

Las pruebas unitarias se corresponden con la prueba de **cada uno de los módulos o clases** del programa de forma independiente y es realizada por el programador en su entorno de trabajo.

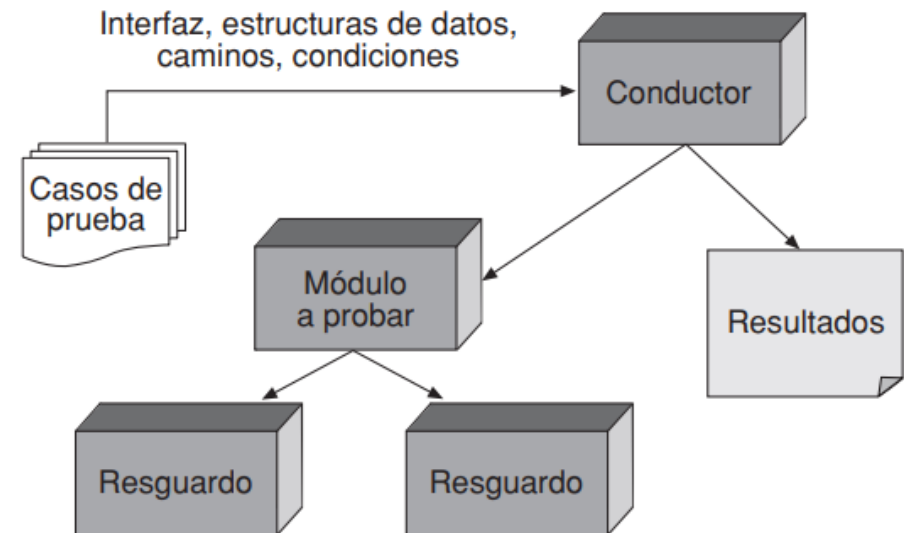
En definitiva, consiste en probar **los bloques más pequeños con identidad propia** presentes dentro del programa. De esta forma, si una prueba descubre un nuevo error, este está más localizado. Además, se pueden probar simultáneamente varios módulos

Pruebas Unitarias

El enfoque de las pruebas unitarias es el de las **técnicas de caja blanca**. Para ello se crean módulos conductores y módulos resguardo.

Un módulo conductor o módulo impulsor es un módulo específicamente creado para la prueba que llama al módulo a probar.

Un módulo resguardo o módulo auxiliar es un módulo específicamente creado para la prueba que es llamado por el módulo a probar.



Pruebas unitarias

Se construyen, por tanto, **módulos resguardo o módulos conductores** cuya función es el paso de parámetros o variables o hacer las llamadas necesarias al módulo que se desea probar de tal forma que se pruebe el módulo de forma unitaria pero con el paso de parámetros real o desde otros módulos.

De esta forma, se prueba el módulo en cuestión y se corrigen los errores que surjan de dicho módulo, de tal manera que cuando se pase a la siguiente etapa de pruebas, los módulos estén todos probados de forma independiente.

Pruebas Unitarias (Principios FIRST)

Principios FIRST

Fast

La ejecución del código de pruebas debe ser rápida. Si las pruebas consumen demasiado tiempo acabaremos por no hacerlas.

Pruebas Unitarias (Principios FIRST)

Independent

Una prueba no puede depender de otras. Cada prueba debe ser unitaria, debe poder realizarse de modo aislado..

Pruebas Unitarias (Principios FIRST)

Repeatable

Las pruebas se deben poder repetir en cualquier momento y la cantidad de veces que sea necesario. El resultado de una prueba debe ser siempre el mismo.

Pruebas Unitarias (Principios FIRST)

Self-validating

Solo hay dos posibles resultados de una prueba: ((La prueba pasó con éxito)) o ((La prueba falló)).

Pruebas Unitarias (Principios FIRST)

Timely

Las pruebas han de escribirse en el momento de escribir el código, y no al final de toda la fase de desarrollo

JUnit

JUnit es un framework que permite la automatización de pruebas unitarias sobre clases desarrolladas en el lenguaje Java. Fue creado en 1997 por **Erich Gamma y Kent Beck** basándose inicialmente en un framework para Smalltalk llamado SUnit y que fue desarrollado por este último.

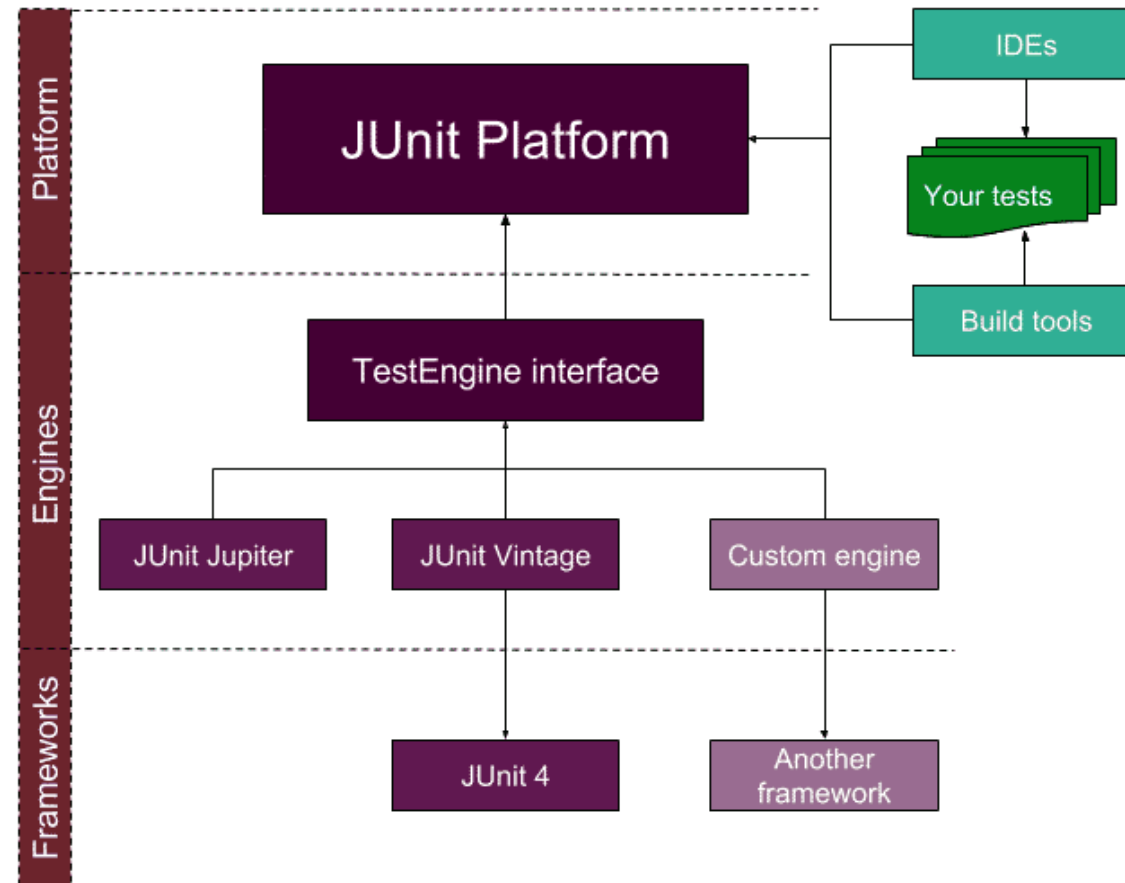
Desde su aparición JUnit ha sido utilizado en multitud de proyectos software como una herramienta capaz de asistir eficazmente al desarrollador software en la realización de pruebas unitarias. Y ya desde hace varios años, JUnit se ha convertido en la herramienta de referencia para la realización de pruebas unitarias en el mundo de desarrollo Java.

JUnit

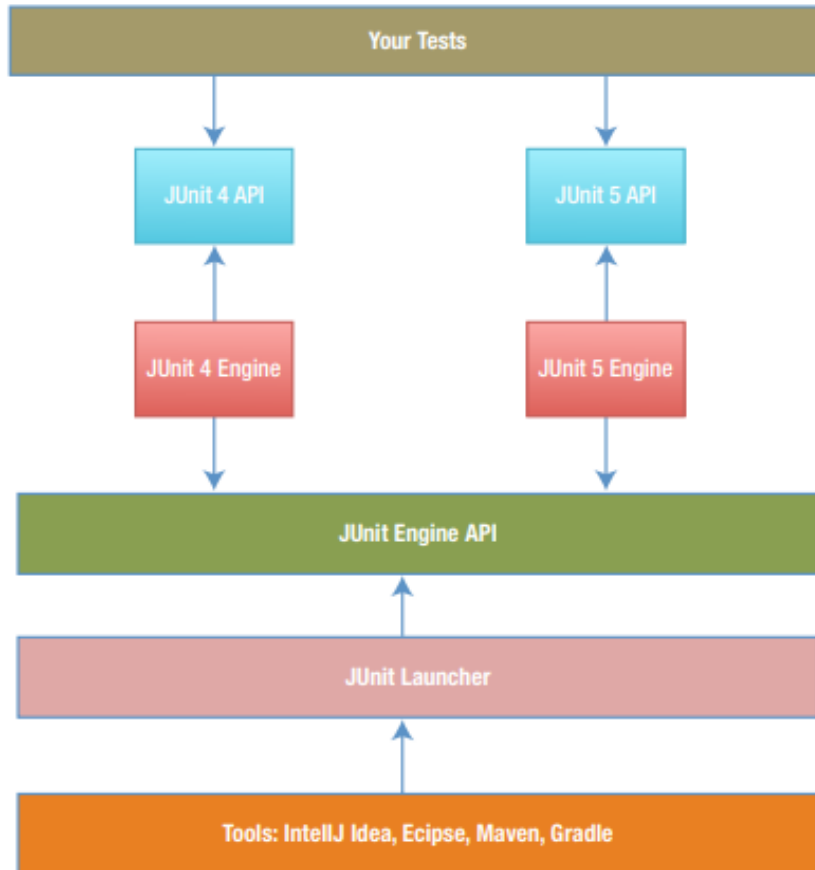
Principales funciones:

- Se encarga de resolver **las tareas repetitivas** asociadas al proceso de pruebas como son la organización de las clases de prueba y el manejo de las situaciones de error.
- **Delimita claramente las tareas** del desarrollador, que se restringen a plasmar la información contenida en los casos de prueba en forma de código Java.
- Proporciona un **conjunto de métodos que facilitan la tarea de comprobación** de las condiciones contenidas en los casos de prueba definidos. Estos métodos en adelante serán llamados métodos assert.
- Proporciona un **mecanismo para ejecutar los casos de prueba** de forma ordenada a la vez que mantiene información en tiempo de ejecución de los defectos software encontrados. Dicha información es mostrada al usuario al final del proceso.
- Consta de un muy reducido número de clases y métodos por lo que **la curva de aprendizaje es bastante plana**. Siendo esta una de las principales razones de la enorme popularidad que ha alcanzado la herramienta.

Pruebas unitarias - Arquitectura



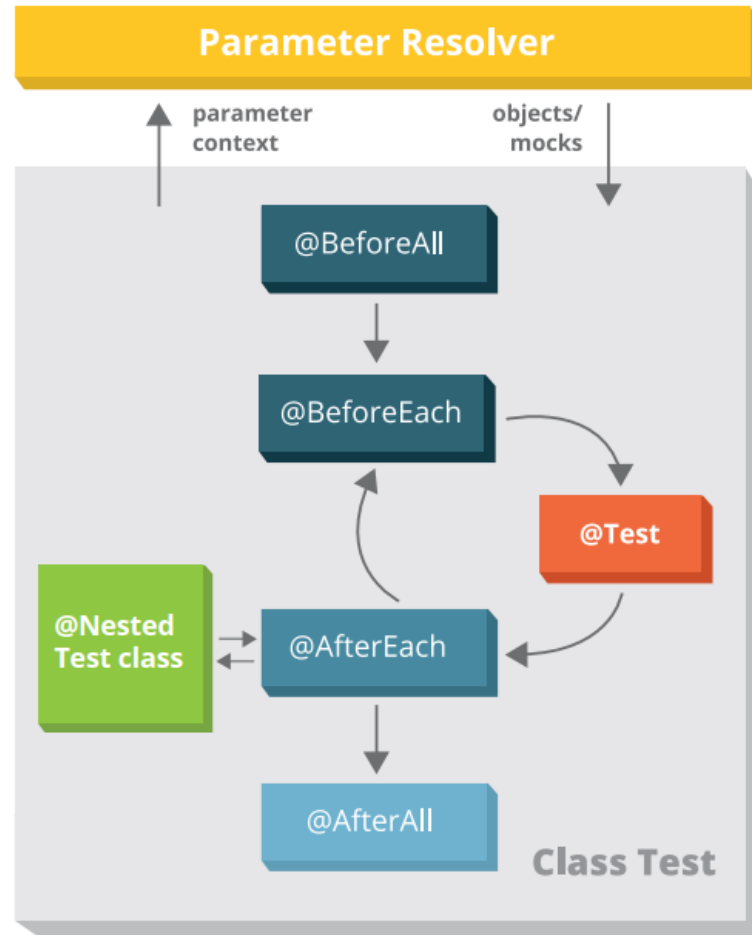
Pruebas unitarias - Arquitectura



- Junit-api
- Junit-platform-launcher
- Junit-platform-engine
- Junit-jupiter-engine
- Junit-vintage-engine
- Junit-platform-commons

JUnit

Lifecycle of standard tests



@Test - marks a test method

@TestFactory - method to create test cases at Runtime

@DisplayName - make reports readable with proper test names

@BeforeAll/@BeforeEach - lifecycle methods executed prior testing

@AfterAll/AfterEach - lifecycle methods for cleanup

@Tag - declare tags to separate tests into suites

@Disabled - make JUnit skip this test.

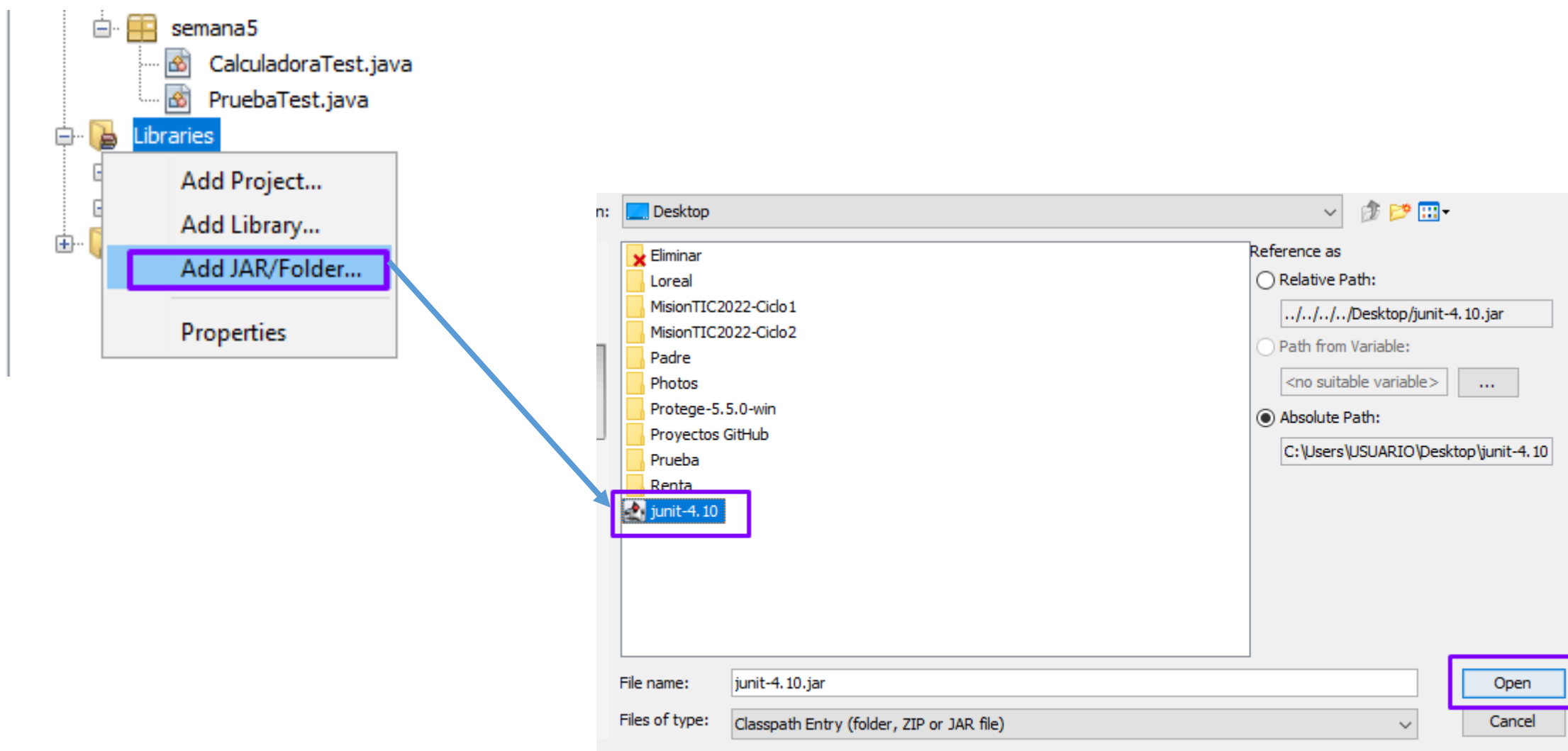
Use **@Nested** on an inner class to control the order of tests.

Lista de Principales métodos

- `assertArrayEquals(expecteds, actuals);`
- `assertEquals(message, expectedResult, result);`
- `assertNotEquals(message, expectedResult, result);`
- `assertFalse(message, true);`
- `assertSame(message, expectedResult, result);`
- `assertFalse(message, true);`
- `assertTrue(message, true);`
- `assertNull(result);`
- `assertNotNull(result);`
- `assertNotSame(message, expectedResult, result);`

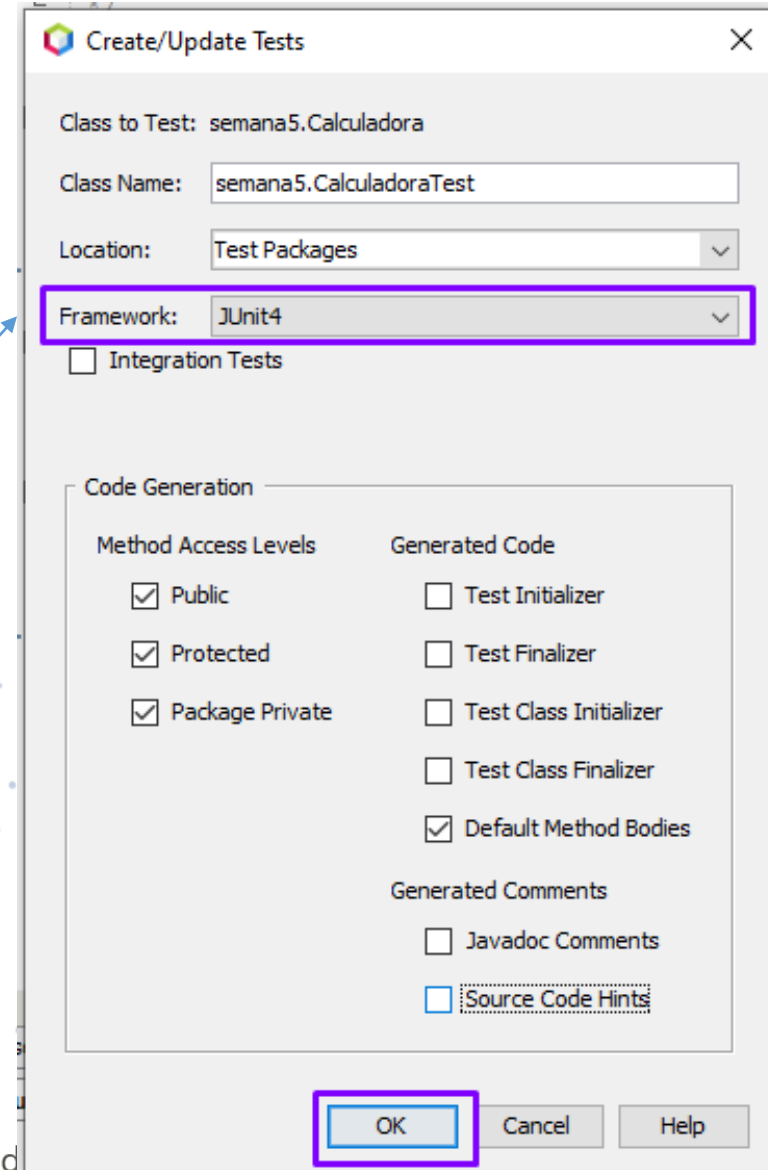
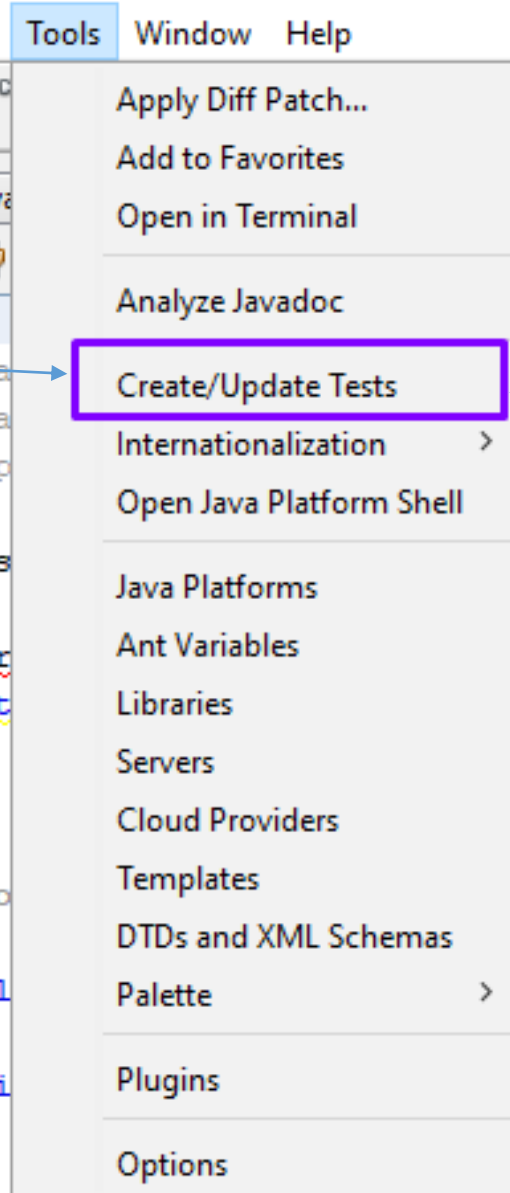
Quizziz

<https://quizizz.com/join?gc=20550118>



OPCION 1

```
public class Calculadora {  
  
    public double getSuma(double a, double b) {  
        return a + b;  
    }  
  
    public double getResta(double a, double b) {  
        return a - b;  
    }  
  
}
```




```
20 public class CalculadoraTest {
21
22     public static Calculadora instance;
23
24     public CalculadoraTest() {
25     }
26
27     @Test
28     public void testGetResta() {
29         assertEquals(1, 1, 0);
30     }
31 }
```

semana5.CalculadoraTest > testGetResta >

Test Results × Output - CodigosPruebasNetbeans (test)

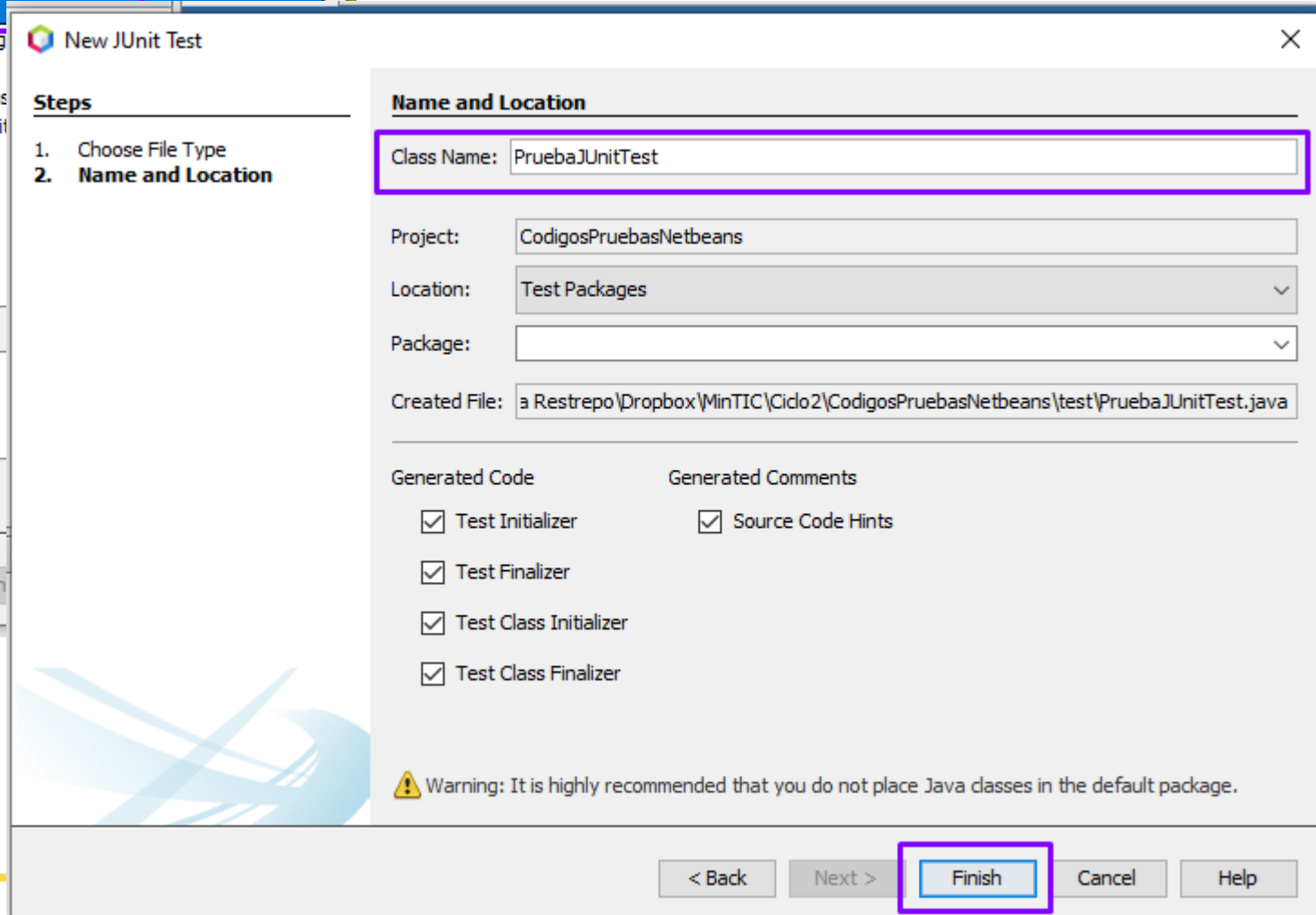
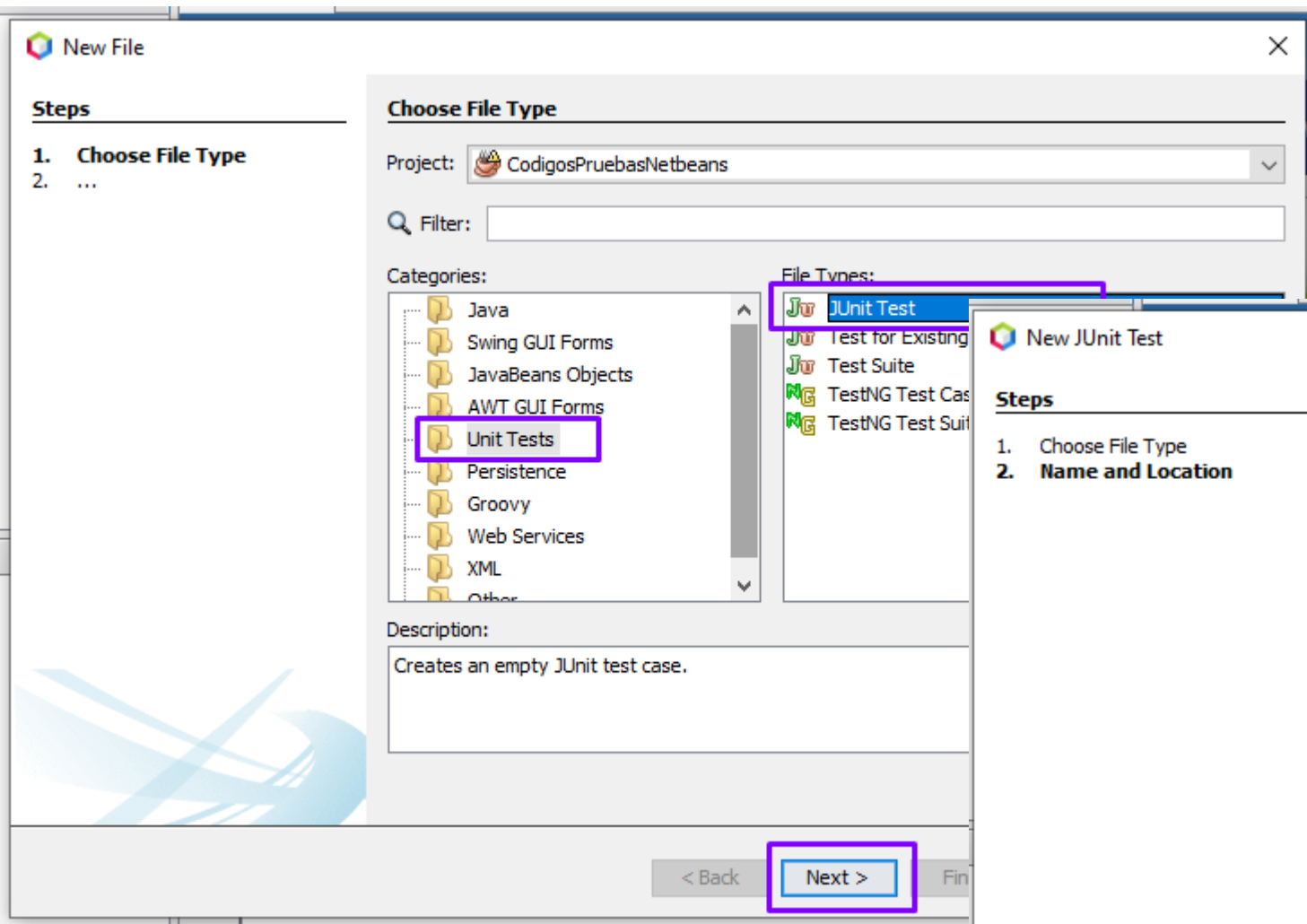
semana5.CalculadoraTest ×

Tests passed: 100.00 %

The test passed. (0.089 s)

- semana5.CalculadoraTest passed
 - testGetResta passed (0.001s)

OPCION 2



New Test for Existing Class

Steps

1. Choose File Type
2. **Existing Class To Test**

Existing Class To Test

Class to Test:

Created Test Class:

Project:

Location:

Created File:

Method Access Levels	Generated Code	Generated Comments
<input checked="" type="checkbox"/> Public	<input type="checkbox"/> Test Initializer	<input checked="" type="checkbox"/> Javadoc Comments
<input checked="" type="checkbox"/> Protected	<input type="checkbox"/> Test Finalizer	<input checked="" type="checkbox"/> Source Code Hints
<input checked="" type="checkbox"/> Package Private	<input checked="" type="checkbox"/> Test Class Initializer	
	<input checked="" type="checkbox"/> Test Class Finalizer	
	<input checked="" type="checkbox"/> Default Method Bodies	

< Back Next > **Finish** Cancel Help

New JUnit Test

Steps

1. Choose File Type
2. **Name and Location**

Name and Location

Class Name:


Project:

Location:

Package:

Created File:

Generated Code	Generated Comments
<input type="checkbox"/> Test Initializer	<input checked="" type="checkbox"/> Source Code Hints
<input type="checkbox"/> Test Finalizer	
<input type="checkbox"/> Test Class Initializer	
<input type="checkbox"/> Test Class Finalizer	

 Warning: It is highly recommended that you do not place Java c

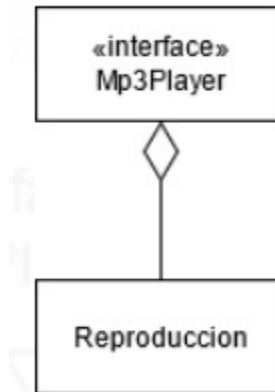
< Back Next > **Finish**

Ejemplos

Operaciones

Ejercicios

- Mp3: Algún participante que lo quiera realizar



- Hacer las pruebas unitarias de la clase cuenta bancaria – variante 2 reto 2

Bases de Datos

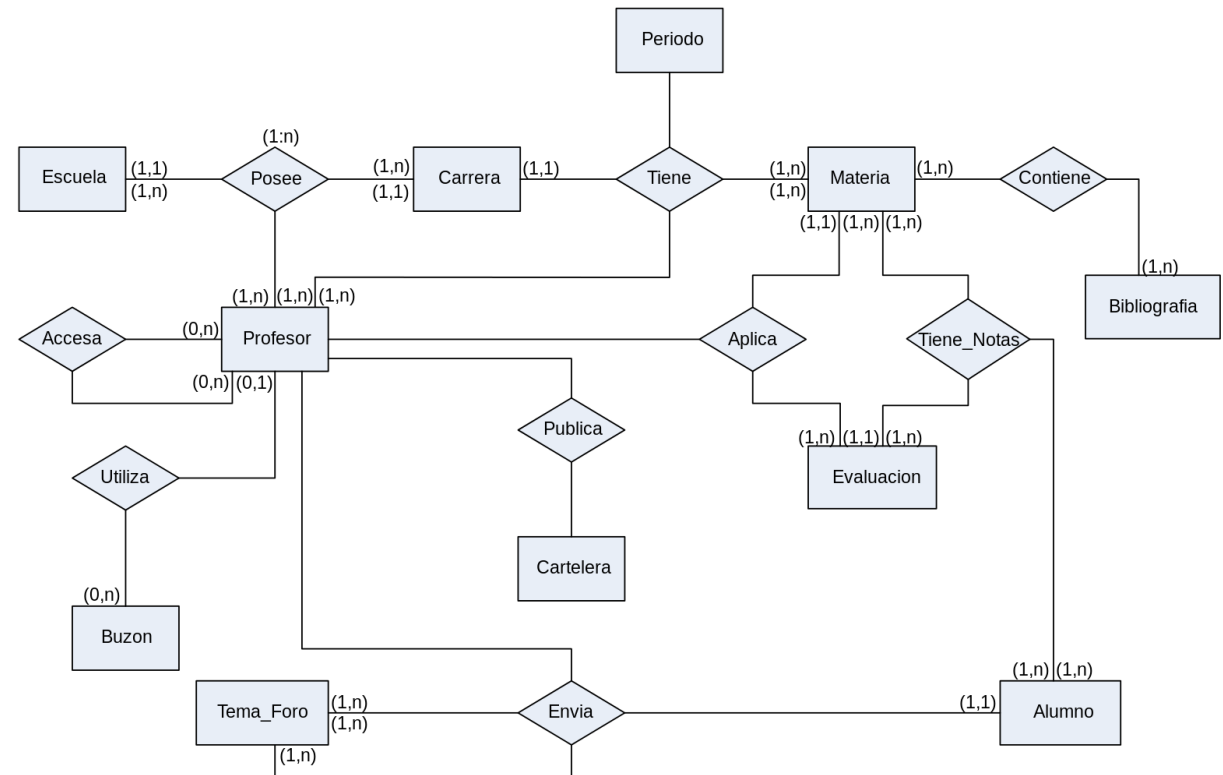
Que es una base de datos

- Una base de datos (db) es una colección organizada de datos, normalmente almacenados en formato electrónico.
- Le permite ingresar, administrar, organizar y recuperar rápidamente información.
- Las bases de datos tradicionales están organizadas por registros (filas), campos (columnas) almacenados en tablas que se almacenan en los archivos de base de datos

Bases de Datos: Diagrama Entidad Relación

Un modelo entidad-relación es una herramienta para el modelo de datos, la cual facilita la representación de entidades de una base de datos.

Fue definido por Peter Chen en 1976.



<https://jorgesanchez.net/manuales/gbd/modelo-relacional.html>

Bases de Datos: Diagrama Entidad Relación

Año	Hecho
1970	Codd publica las bases del modelo relacional
1971-72	Primeros desarrollos teóricos
1973-78	Primeros prototipos de base de datos relacional. Son el System R de IBM. En ese sistema se desarrolla Sequel que con el tiempo cambiará su nombre a SQL.
1974	La Universidad de Berkeley desarrolla Ingres , SGBD relacional basado en cálculo relacional. Utilizaba el lenguaje Quel desarrollado en las universidades y muy popular en la época en ámbitos académicos.
1978	Aparece el lenguaje QBE (<i>Query By Example</i>) lenguaje de acceso relacional a los archivos VSAM de IBM
1979	Aparece Oracle , el primer SGBD comercial relacional (ganando en unas semanas al System/38 de IBM). Implementa SQL y se convertirá en el sistema gestor de bases de datos relacionales líder del mercado. Codd revisa su modelo relacional y lanza el modelo RM/T como un intento de subsanar sus deficiencias.
1981	Aparece Informix como SGBD relacional para Unix
1983	Aparece DB2 , el sistema gestor de bases de datos relacionales de IBM
1984	Aparece la base de datos Sybase que llegó a ser la segunda más popular (tras Oracle)
1986	ANSI normaliza el SQL (SQL/ANSI). SQL es ya de hecho el lenguaje principal de gestión de bases de datos relacionales.
1987	ISO también normaliza SQL. Es el SQL ISO(9075)
1988	La versión 6 de Oracle incorpora el lenguaje procedimental PL/SQL

<https://jorgesanchez.net/manuales/gbd/modelo-relacional.html>

Bases de Datos: Diagrama Entidad Relación

1989	ISO revisa el estándar y publica el estándar SQL Addendum . Microsoft y Sybase desarrollan SQL Server para el sistema operativo OS/2 de Microsoft e IBM . Durante años Sybase y SQL Server fueron el mismo producto.
1990	Versión dos del modelo relacional (RM/V2) realizada por Codd. Propuesta de Michael Stonebraker para añadir al modelo relacional capacidades de orientación a objetos.
1992	ISO publica el estándar SQL 92 (todavía el más utilizado)
1995	Manifiesto de Darwen y Date en el que animan a reinterpretar el modelo relacional desde una perspectiva de objetos. Aparece el modelo objeto/relacional. Aparece MySQL una base de datos relacional de código abierto con licencia GNU que se hace muy popular entre los desarrolladores de páginas web.
1996	ANSI normaliza el lenguaje procedimental basado en SQL y lo llaman SQL/PSM . Permite técnicas propias de los lenguajes de programación estructurada. Aparece el SGBD abierto PostgreSQL como remodelación de la antigua Ingres, utilizando de forma nativa el lenguaje SQL (en lugar de Quel).
1999	ISO publica un nuevo estándar que incluye características más avanzadas. Se llama SQL 99 (también se le conoce como SQL 2000)
2000	Richard Hipp diseña SQLite base de datos relacional que ocupa muy poco, pero que ofrece prestaciones propias de sistemas más grandes. Es muy utilizada en aplicaciones, especialmente en los dispositivos móviles.
2003	ISO publica el estándar SQL 2003 . En él se añade SQL/PSM al estándar.
2006	Estándar ISO. SQL 2006
2008	Estándar ISO. SQL 2008
2011	Estándar ISO. SQL 2011

<https://jorgesanchez.net/manuales/gbd/modelo-relacional.html>

Bases de Datos: Diagrama Entidad Relación

Entidad

Representa una "cosa", "objeto" o "concepto" del mundo real con existencia independiente, es decir, se diferencia únicamente de otro objeto o cosa, incluso siendo del mismo tipo, o una misma entidad.

Atributos

Los atributos son las características que definen o identifican a una entidad. Estas pueden ser muchas, y el diseñador solo utiliza o implementa las que considere más relevantes.

En un conjunto de entidades del mismo tipo, cada entidad tiene valores específicos asignados para cada uno de sus atributos, de esta forma, es posible su identificación unívoca.

Bases de Datos: Diagrama Entidad Relación

Conjunto de relaciones

Consiste en una colección, o conjunto, de relaciones de la misma naturaleza.

Claves

Es un subconjunto del conjunto de atributos comunes en una colección de entidades, que permite identificar inequívocamente cada una de las entidades pertenecientes a dicha colección. Asimismo, permiten distinguir entre sí las relaciones de un conjunto de relaciones.

Bases de Datos: Diagrama Entidad Relación

Cardinalidad de las relaciones

Cardinalidad es el número de entidades con la cual otra entidad puede asociar mediante una relación binaria; la cardinalidad puede ser: Uno a uno, uno a muchos o muchos a uno y muchos a muchos. El tipo de cardinalidad se representa mediante una etiqueta en el exterior de la relación, respectivamente: "1:1", "1:N" y "N:M", aunque la notación depende del lenguaje utilizado, la que más se usa actualmente es el unificado. Otra forma de expresar la cardinalidad es situando un símbolo cerca de la línea que conecta una entidad con una relación:

"0" si cada instancia de la entidad no está obligada a participar en la relación.

"1" si toda instancia de la entidad está obligada a participar en la relación y, además, solamente participa una vez.

"N" , "M", ó "*" si cada instancia de la entidad no está obligada a participar en la relación y puede hacerlo cualquier número de veces.

Integridad referencial

La integridad referencial (RI) es un concepto de base de datos que se utiliza para garantizar que las relaciones entre las tablas de base de datos permanecen sincronizadas durante las modificaciones de datos.



PRIMARY KEY



FOREIGN KEY

Clave Primaria

Un concepto importante del diseño de una tabla de base de datos es el uso de una **CLAVE PRIMARIA**, un atributo o conjunto de atributos que se utiliza para identificar de forma única cada fila.

- Una tabla solo puede tener una clave principal que se crea mediante una restricción de clave principal y se aplica mediante la creación de un índice único en las columnas de clave principal
- Una columna que participa en la restricción de clave principal no puede aceptar valores NULL.



PRIMARY KEY

Clave Foránea

Foreign Key es una columna o combinación de columnas que se utilizan para establecer un vínculo entre los datos de dos tablas. Las columnas utilizadas para crear la clave principal en una tabla también se utilizan para crear la restricción de clave externa y se pueden utilizar para hacer referencia a datos de la misma tabla o de otra tabla.

- Una clave externa no tiene que hacer referencia a una clave principal, se puede definir para hacer referencia a una restricción única en la misma tabla o en otra tabla

- Una columna que participa en la restricción de clave externa puede aceptar valores NULL, pero si contiene un valor NULL, se omite el proceso de comprobación.



FOREIGN KEY

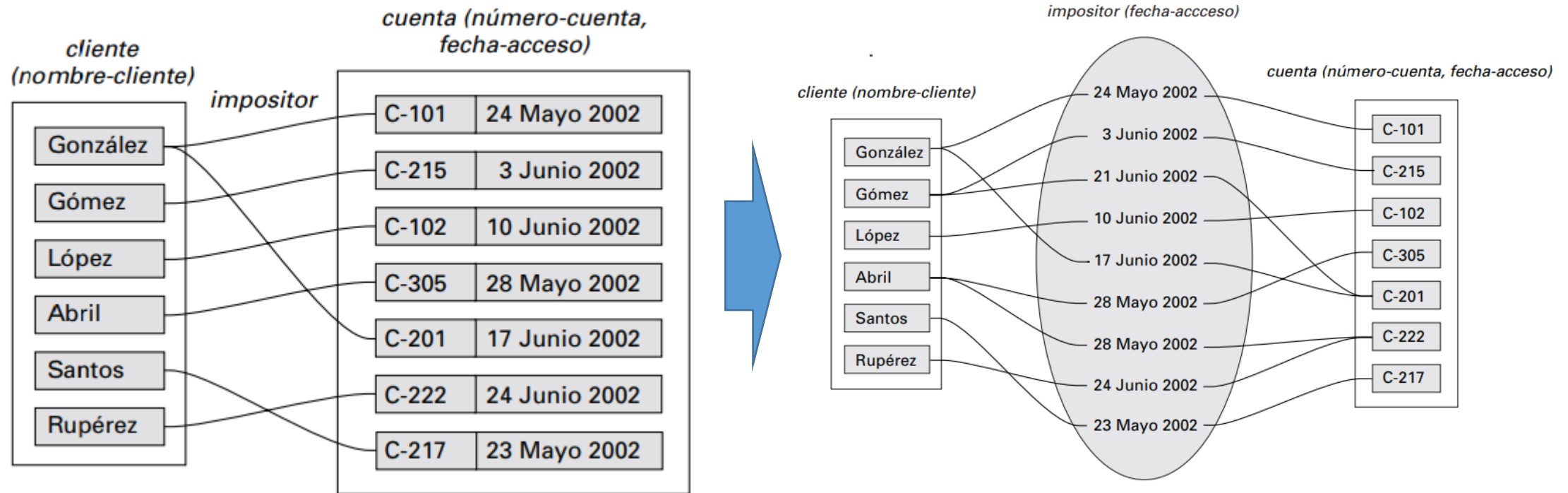
Clave Foránea

- Restricción **PRIMARY KEY**: un atributo o conjunto de atributos que se utiliza para identificar de forma única cada fila.
- Restricción **FOREIGN KEY**: una columna o combinación de columnas que se usa para establecer un vínculo entre los datos de dos tablas
- Restricción **UNIQUE**: permite exigir la unicidad en columnas distintas de la clave principal
- **UNIQUE INDEX**: garantiza que la clave de índice no contiene valores duplicados y que cada fila de la tabla o vista es única de alguna manera
- **TRIGGERS**: sentencias T-SQL complejas utilizadas para proporcionar integridad de datos cuando se modifican los datos de la tabla



FOREIGN KEY

Clave Foránea



SQL

SQL (por sus siglas en inglés **Structured Query Language**; en español lenguaje de consulta estructurada) es un lenguaje de dominio específico utilizado en programación, diseñado para administrar, y recuperar información de sistemas de gestión de bases de datos relacionales.

Una de sus principales características es el manejo del álgebra y el cálculo relacional para efectuar consultas con el fin de recuperar, de forma sencilla, información de bases de datos, así como realizar cambios en ellas.

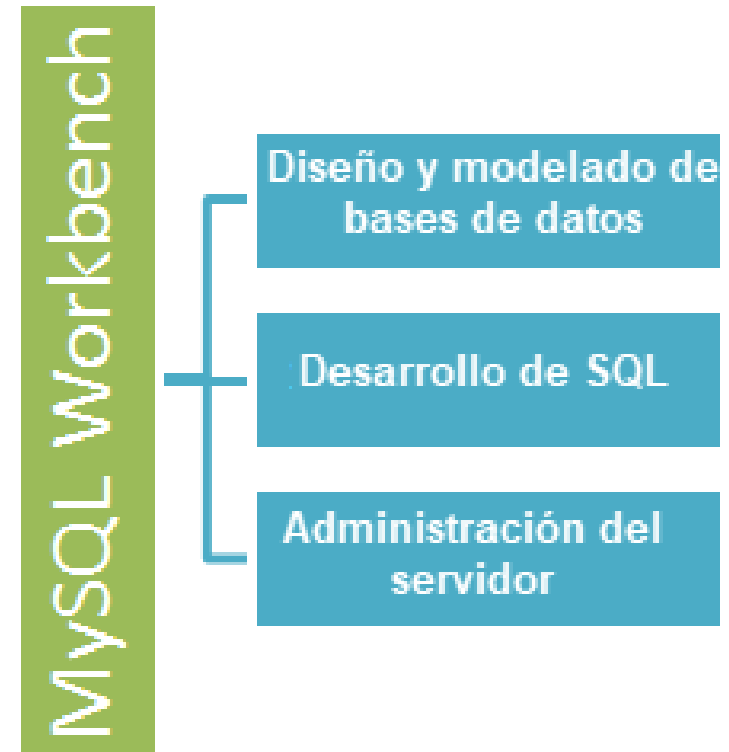
SQL fue uno de los primeros lenguajes comerciales para el modelo relacional de Edgar Frank Codd como se describió en su artículo de investigación de 1970 El modelo relacional de datos para grandes bancos de datos compartidos. A pesar de no adherirse totalmente al modelo relacional descrito por Codd, pasó a ser el lenguaje de base de datos más usado.

Herramientas

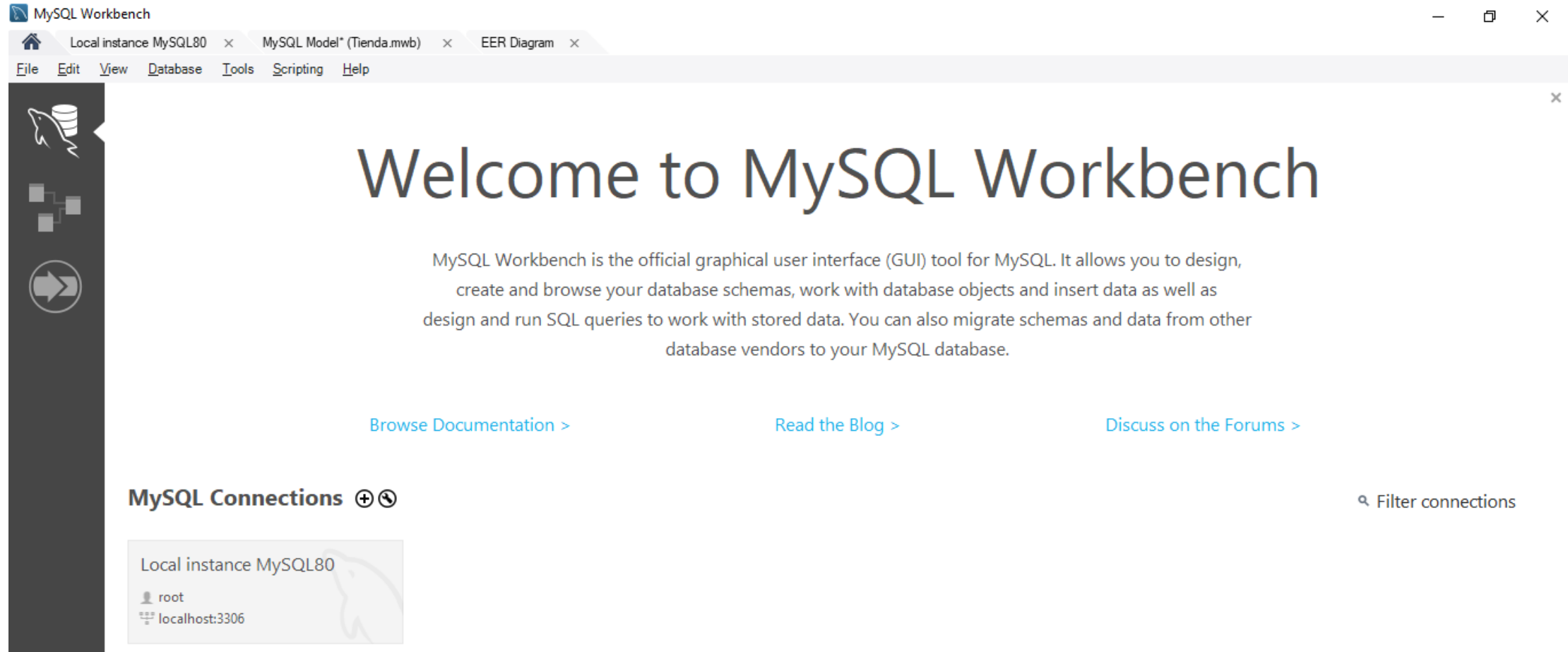
- MySQL Workbench
- MySQL Server o XAMPP
- Controlador para la implementación de la conexión de bases de datos (Connector/J), oficial JDBC para MySQL

MySQL Workbench

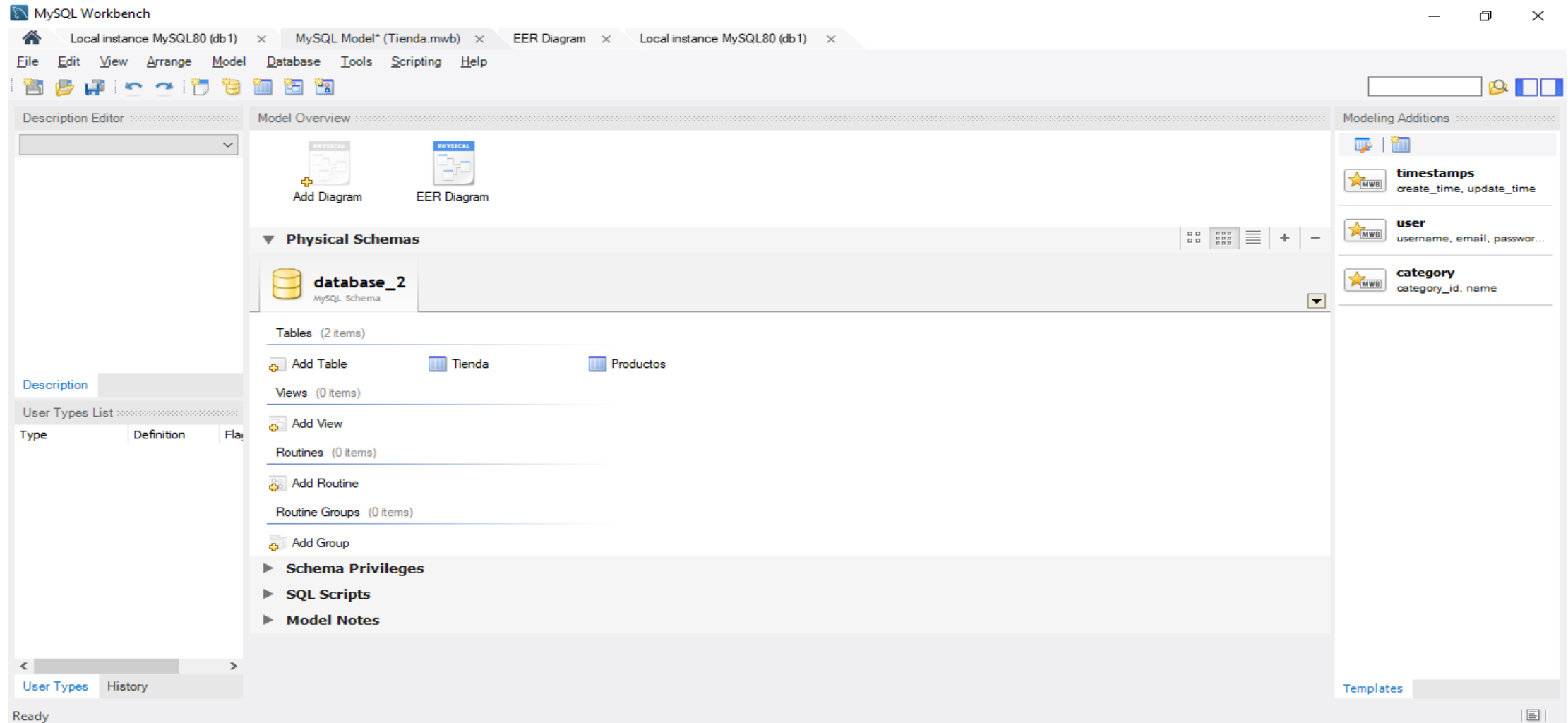
Herramienta de **diseño y modelado de bases de datos visuales** para bases de datos relacionales de servidores MySQL



MySQL Workbench



MySQL Workbench



MySQL Installer

<https://dev.mysql.com/downloads/mysql/>

MySQL Community Downloads

MySQL Installer

General Availability (GA) Releases Archives

MySQL Installer 8.0.26

Select Operating System:
Microsoft Windows

Looking for previous GA versions?

Windows (x86, 32-bit), MSI Installer (mysql-installer-web-community-8.0.26.0.msi)	8.0.26	2.4M	Download
Windows (x86, 32-bit), MSI Installer (mysql-installer-community-8.0.26.0.msi)	8.0.26	450.7M	Download

We suggest that you use the MD5 checksums and GnuPG signatures to verify the integrity of the

MySQL Community Downloads

Login Now or Sign Up for a free account.

An Oracle Web Account provides you with the following advantages:

- Fast access to MySQL software downloads
- Download technical White Papers and Presentations
- Post messages in the MySQL Discussion Forums
- Report and track bugs in the MySQL bug system

Login »

using my Oracle Web account

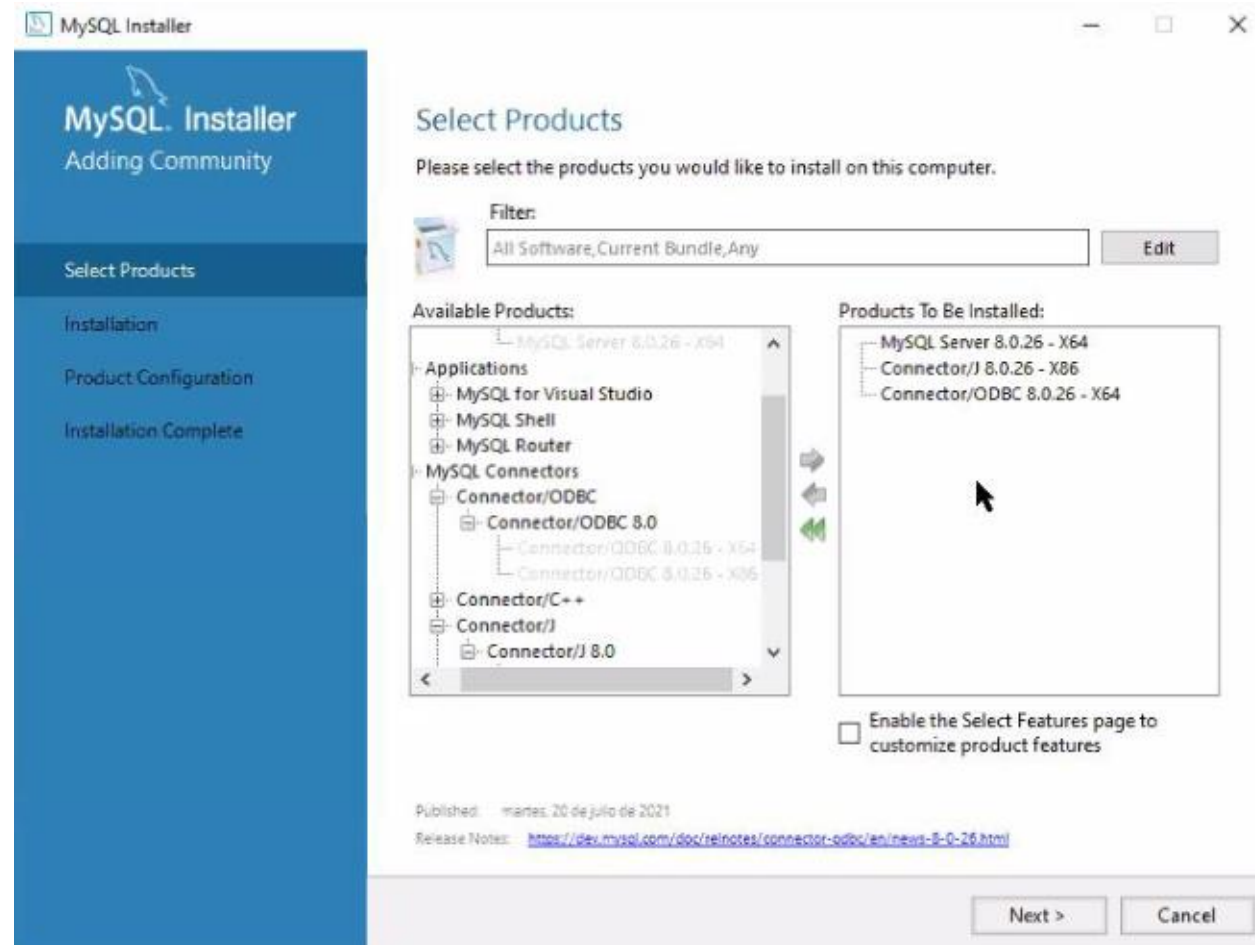
Sign Up »

for an Oracle Web account

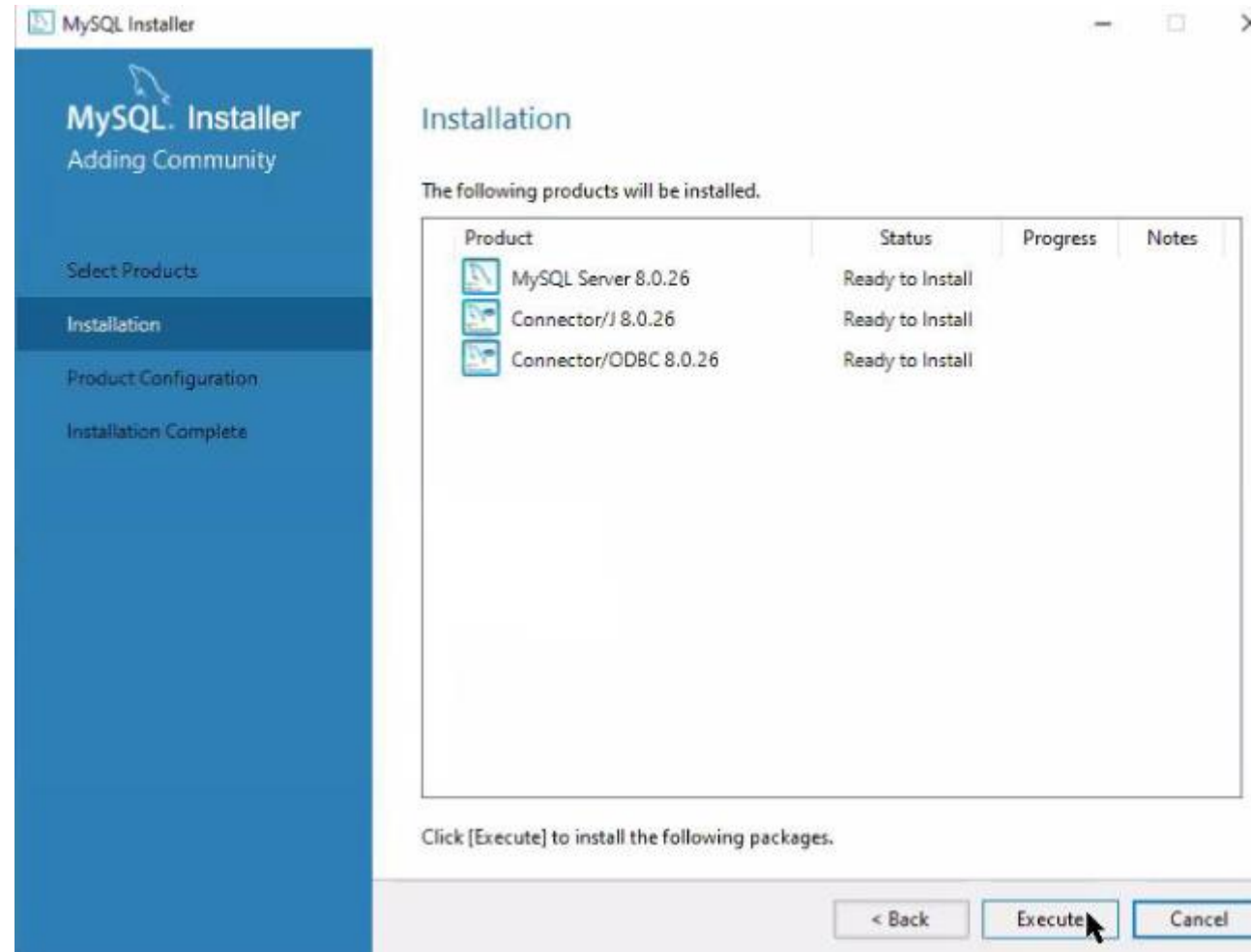
MySQL.com is using Oracle SSO for authentication. If you already have an Oracle Web account, click the Login link. Otherwise, you can signup for a free account by clicking the Sign Up link and following the instructions.

No thanks, just start my download.

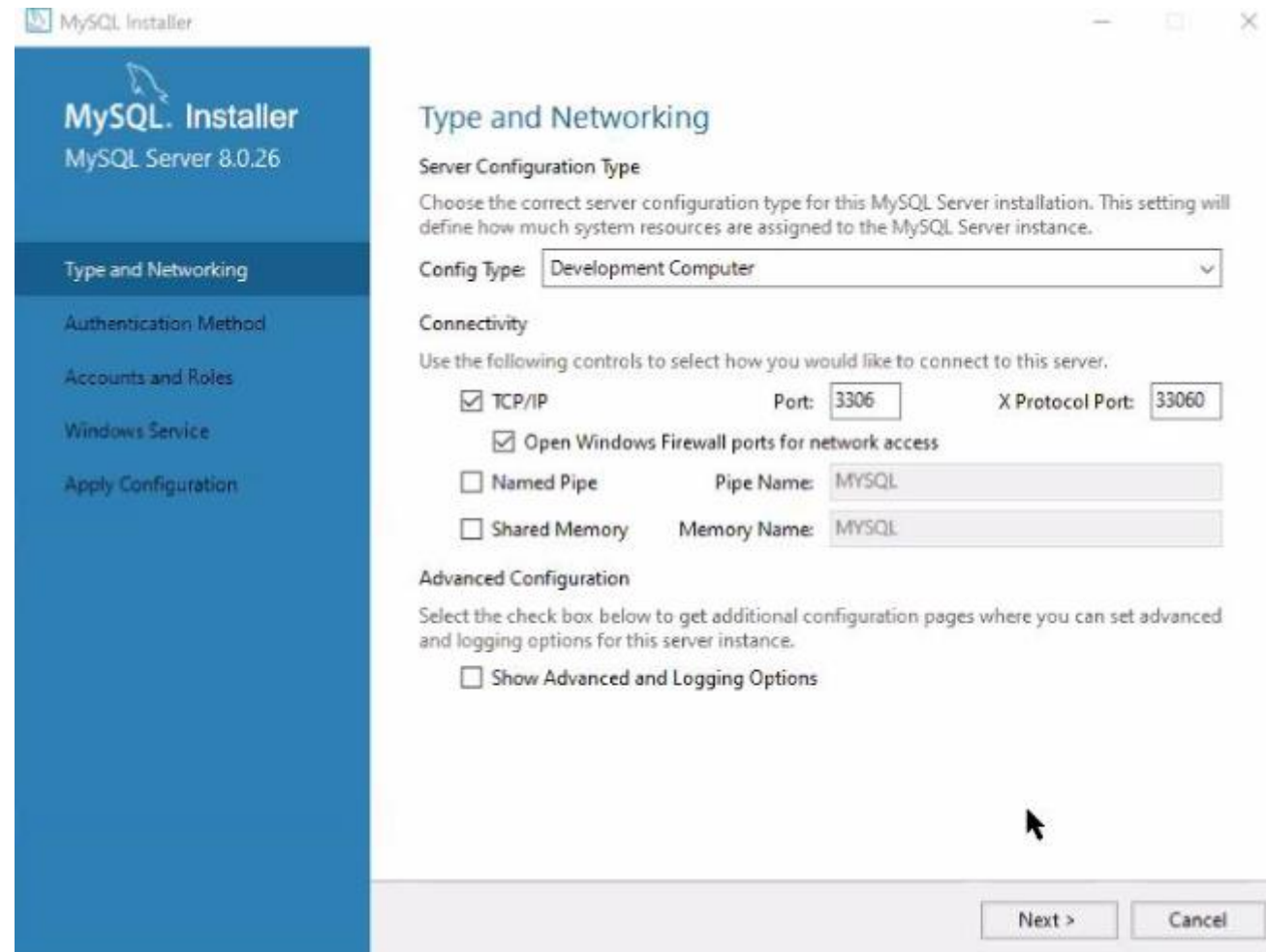
MySQL Installer



MySQL Installer



MySQL Installer



The screenshot shows the 'MySQL Installer' window for 'MySQL Server 8.0.26'. The left sidebar contains a list of configuration steps: 'Type and Networking' (selected), 'Authentication Method', 'Accounts and Roles', 'Windows Service', and 'Apply Configuration'. The main area is titled 'Type and Networking' and contains the following sections:

- Server Configuration Type**
Choose the correct server configuration type for this MySQL Server installation. This setting will define how much system resources are assigned to the MySQL Server instance.
Config Type:
- Connectivity**
Use the following controls to select how you would like to connect to this server.
 - ☒ TCP/IP Port: X Protocol Port:
 - ☒ Open Windows Firewall ports for network access
 - ☐ Named Pipe Pipe Name:
 - ☐ Shared Memory Memory Name:
- Advanced Configuration**
Select the check box below to get additional configuration pages where you can set advanced and logging options for this server instance.
 - ☐ Show Advanced and Logging Options

At the bottom right, there are 'Next >' and 'Cancel' buttons.

MySQL Installer

MySQL Installer

MySQL Server 8.0.26

Type and Networking

Authentication Method

Accounts and Roles

Windows Service

Apply Configuration

Accounts and Roles

Root Account Password
Enter the password for the root account. Please remember to store this password in a secure place.

MySQL Root Password:

Repeat Password:

MySQL User Accounts
Create MySQL user accounts for your users and applications. Assign a role to the user that consists of a set of privileges.

MySQL User Name	Host	User Role
-----------------	------	-----------

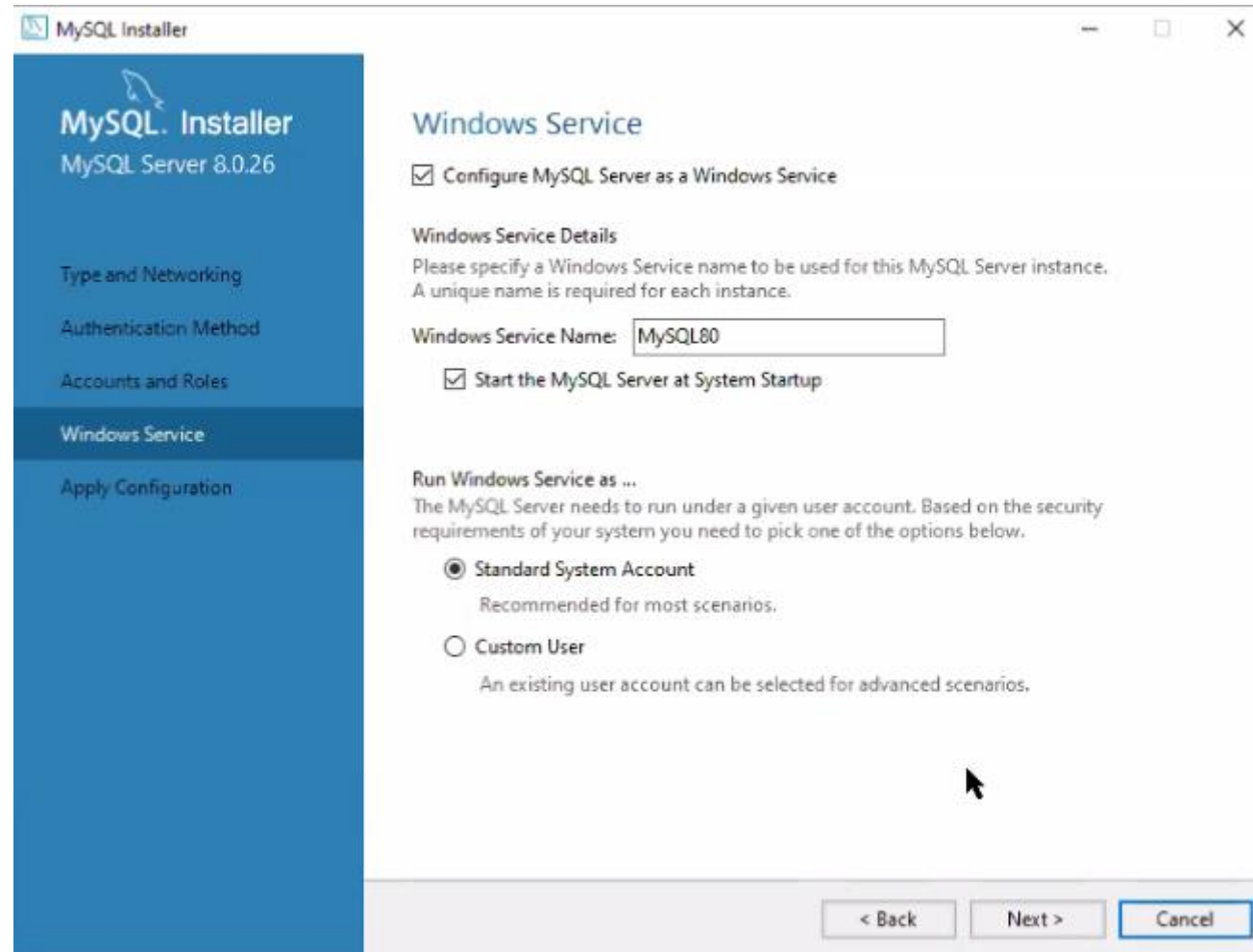
Add User

Edit User

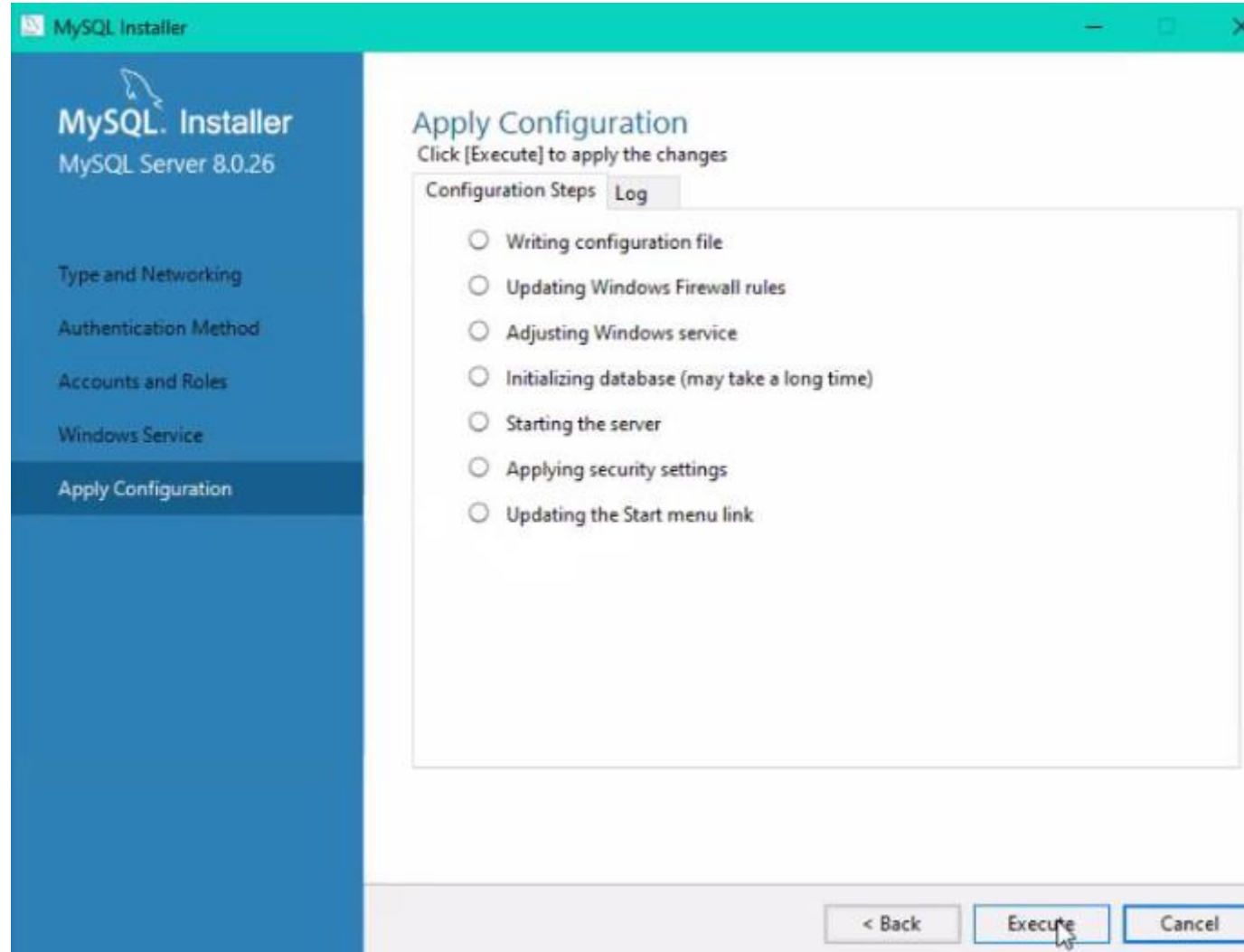
Delete

< Back Next > Cancel

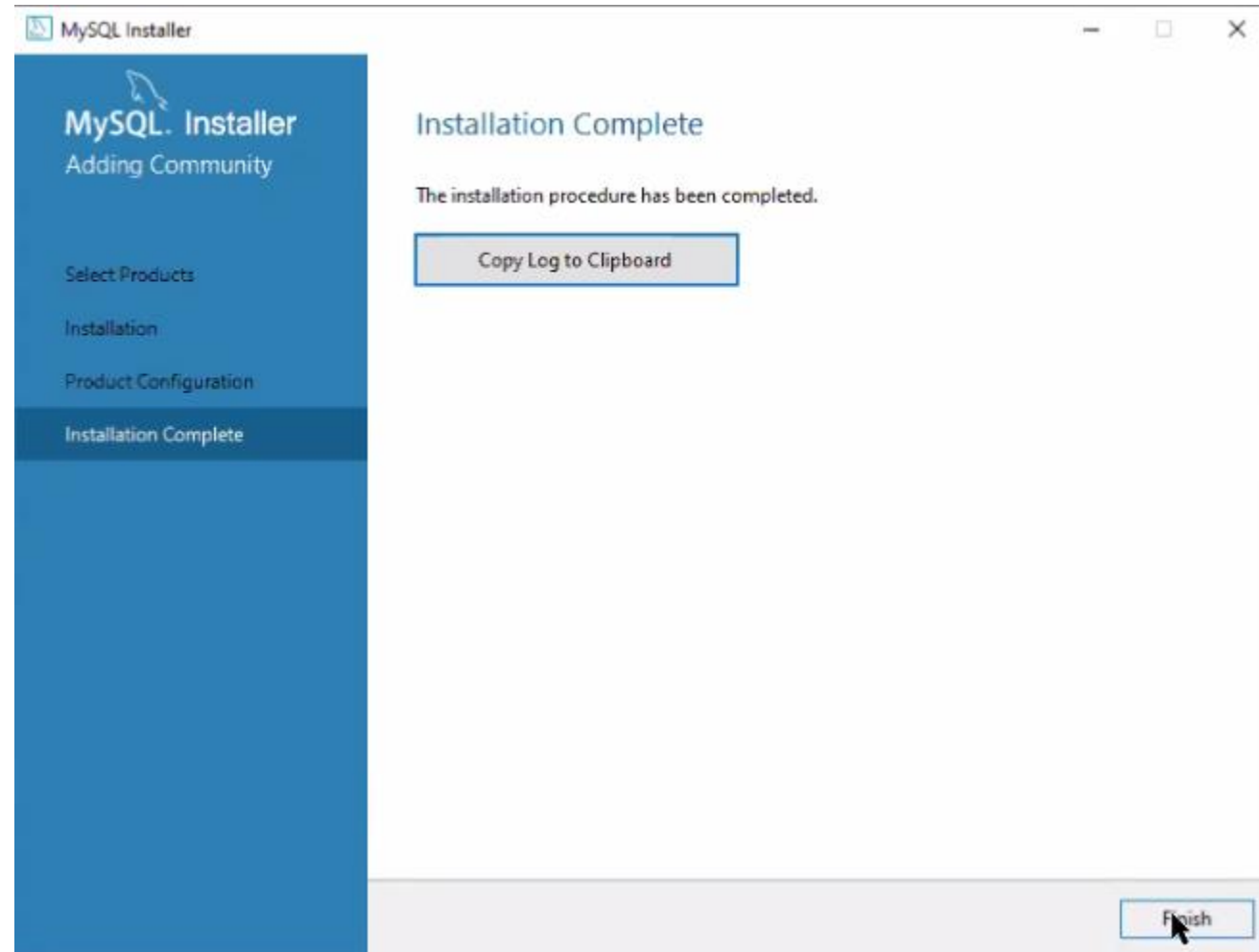
MySQL Installer



MySQL Installer

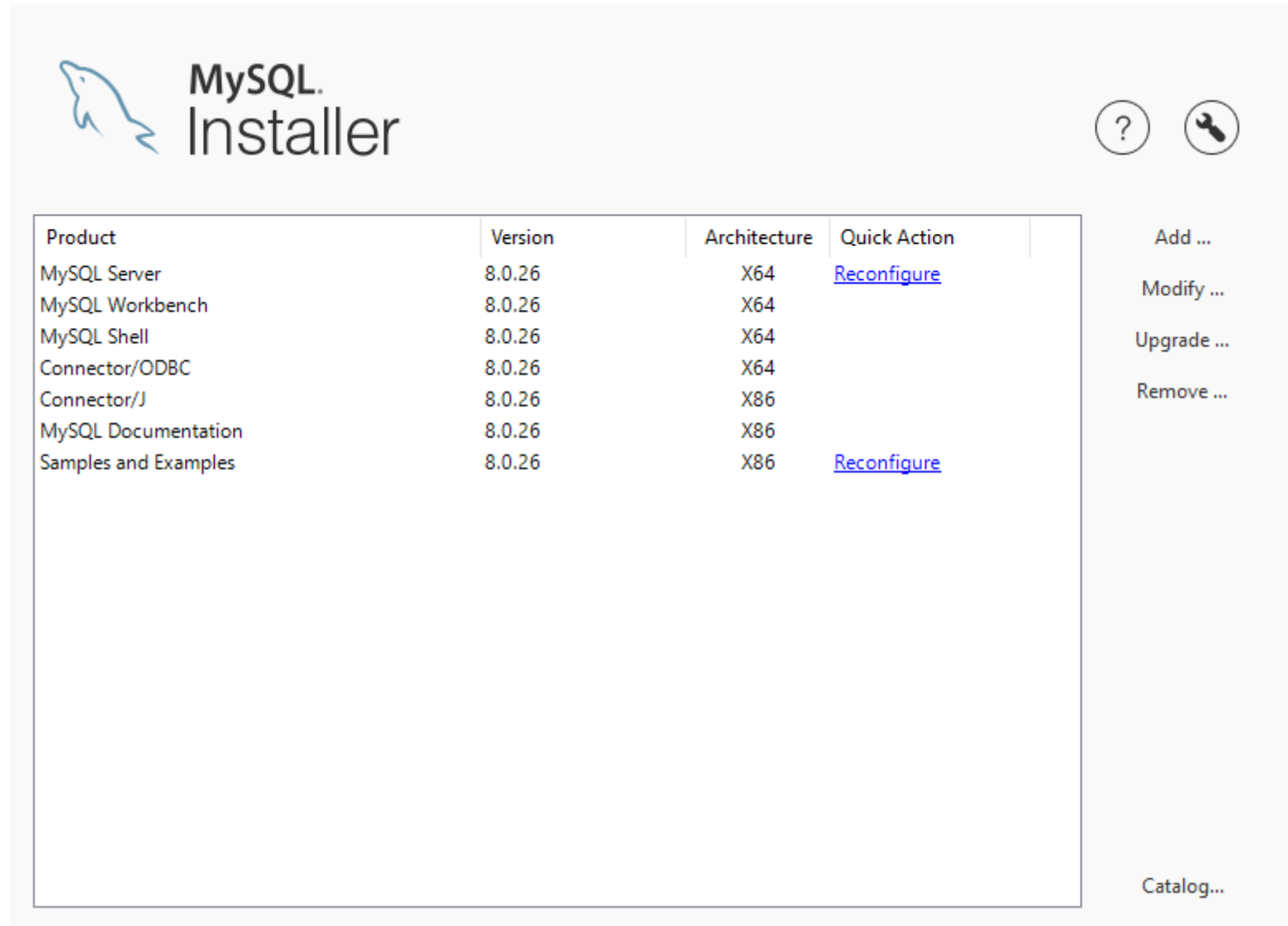


MySQL Installer



MySQL

MySQL Installer





The MySQL Installer window displays a list of installed products. The window title is "MySQL Installer". The main content area contains a table with the following data:

Product	Version	Architecture	Quick Action
MySQL Server	8.0.26	X64	Reconfigure
MySQL Workbench	8.0.26	X64	
MySQL Shell	8.0.26	X64	
Connector/ODBC	8.0.26	X64	
Connector/J	8.0.26	X86	
MySQL Documentation	8.0.26	X86	
Samples and Examples	8.0.26	X86	Reconfigure

On the right side of the window, there are several buttons: "Add ...", "Modify ...", "Upgrade ...", "Remove ...", and "Catalog...". There are also icons for help (?) and settings (wrench) in the top right corner.

MySQL Connections

MySQL Connections  

Filter connections

Local instance MySQL80

root

localhost:3306

Setup New Connection

Connection Name: Type a name for this connection

Connection Method: Standard (TCP/IP) Method to use

Parameters SSL Advanced

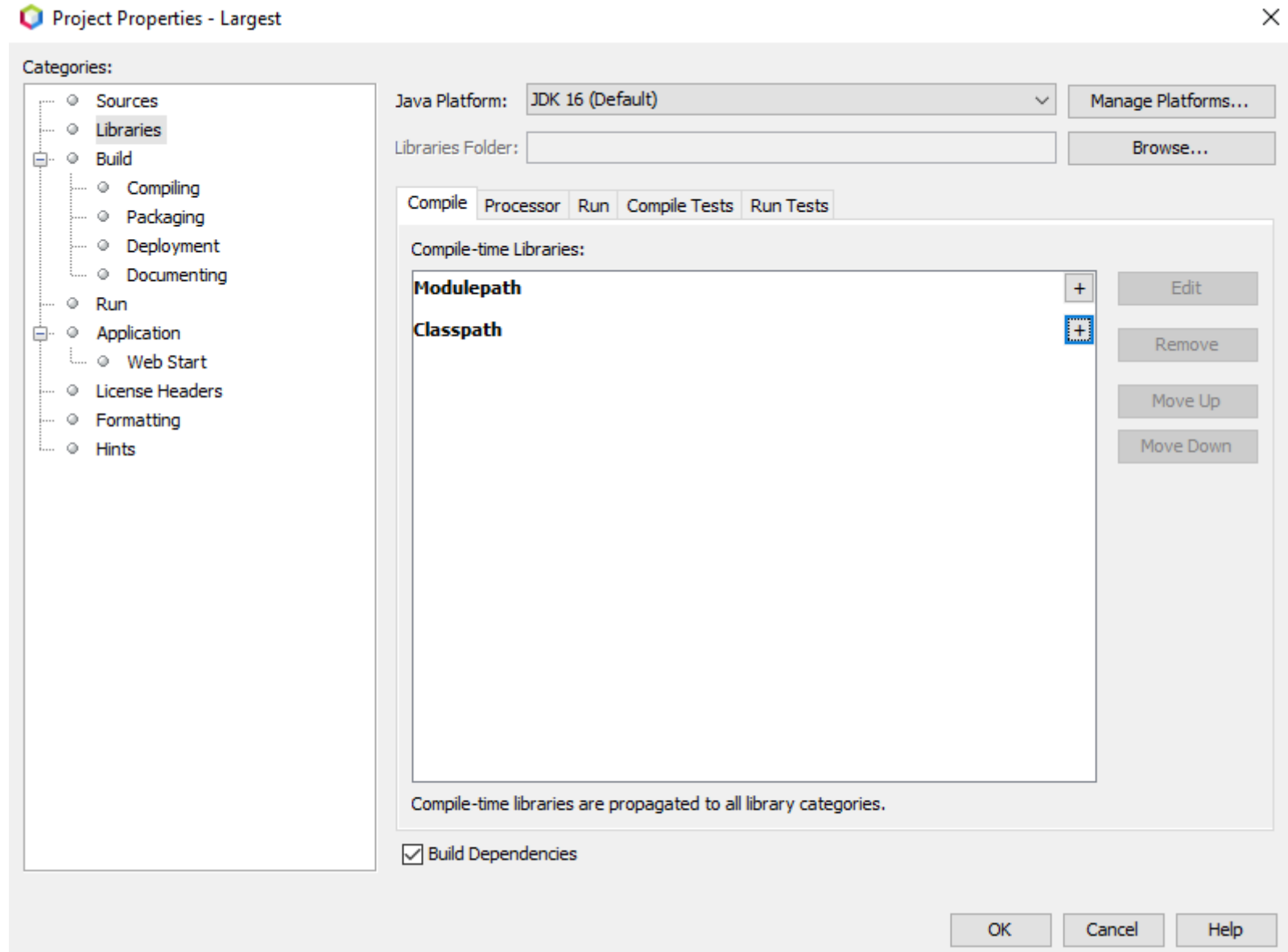
Hostname: Port: Name or IP address of the server. For TCP/IP connections, the port must be specified.

Username: Name of the user to connect to the database.

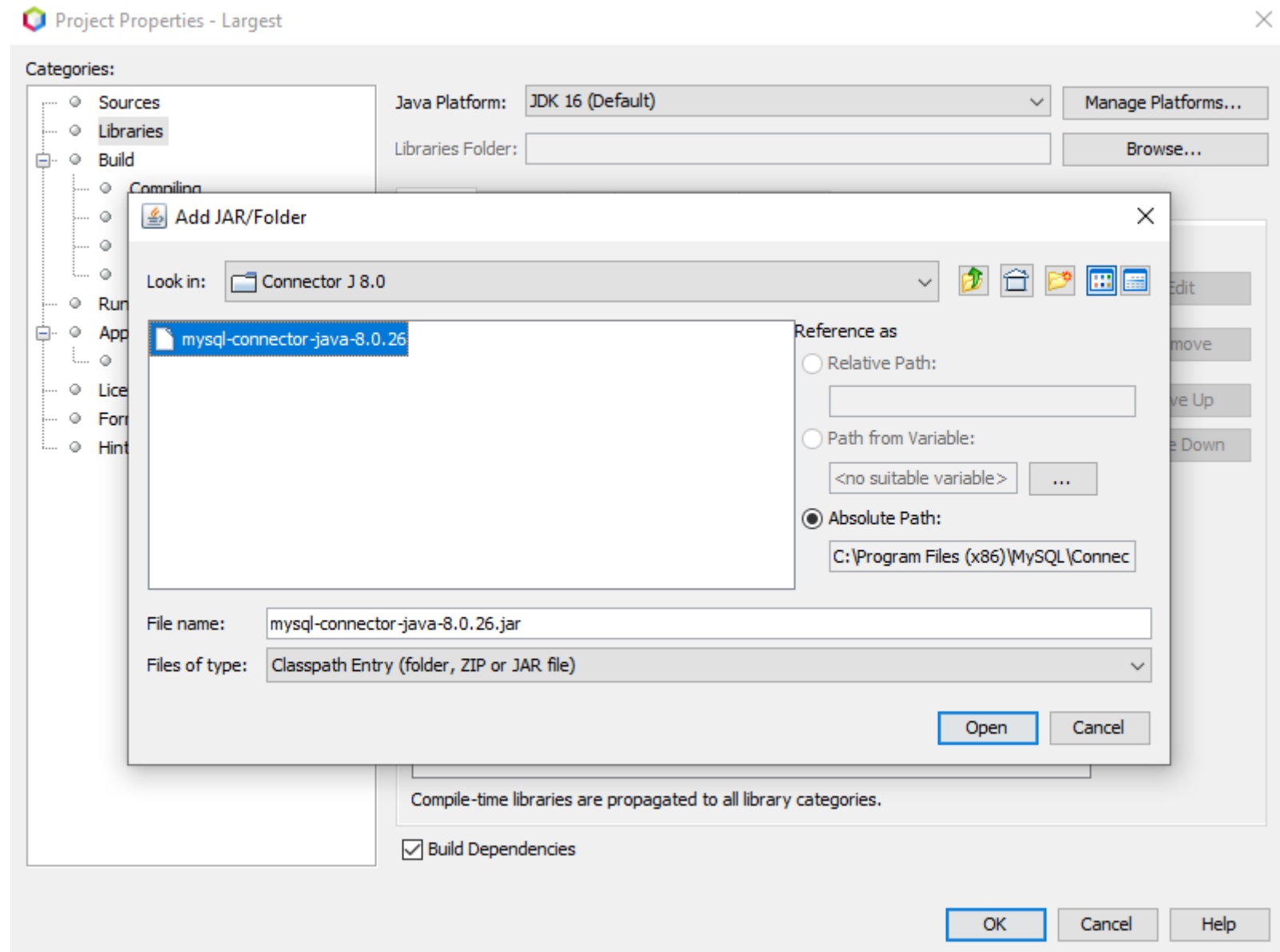
Password: Store in Vault ... Clear The user's password. Will be stored in the vault if the checkbox is checked. If not set, the user will be prompted for the password.

Default Schema: The schema to use as default. If blank, the user will be prompted to select it later.

Connector/J



Connector/J



Normalización

Proceso de optimización de la base de datos que incluye la creación de tablas y el establecimiento de relaciones, esto permite

- Almacenamiento con el menor espacio posible
- Eliminar datos repetidos
- Eliminar errores lógicos
- Datos ordenados

Niveles de normalización



1FN: Eliminar grupos repetidos

2FN: Eliminar datos redundantes

3FN: Eliminar columnas que no dependan de la clave

4FN: Aislar relaciones múltiples independientes

5FN: Aislar relaciones múltiples semánticamente relacionadas

Ejemplo de normalización

Matrícula	Nombre	Dirección	Teléfono	Materia	Num Materia	Carrera
1	Sergio	Puebla 22	56565656	Base de datos	123	Sistemas
1	Sergio	Puebla 22	56565656	Programación web	234	Sistemas
1	Sergio	Puebla 22	56565656	Programación visual	231	Sistemas
2	Ana	Reforma 1	23232323	Base de datos	123	Sistemas

Ejemplo de normalización

Tercer Forma Normal

Matrícula	Nombre	Dirección	Teléfono	No Carrera
1	Sergio	Puebla 22	56565656	1234
2	Ana	Reforma 1	23232323	1234

No Carrera	Carrera
1234	Sistemas
6789	Mecatrónica

Matrícula	Num Materia
1	123
1	234
1	234
2	123

Materia	Num Materia
Base de datos	123
Programación web	234
Programación visual	234
Base de datos	123

Ejemplo de normalización

Ejercicio servicios

Ejemplo de entidad Relación

Biblioteca

Ejercicio de entidad Relación

Realizar el esquema entidad relación para la tienda, en workbench