

IISSI-1 Prueba de Laboratorio.	Curso 2022-23 Diciembre 2022
-----------------------------------	---------------------------------

Instrucciones

- Descargue el proyecto del examen del siguiente repositorio: <https://github.eii.us.es/IISSI-2223/examen-lab-iissi.git>.
- Prepare el proyecto creando la BD ModeloExamen en HeidiSQL, revisando settings.py, y ejecutando "silence createdb", "silence createapi", "silence createtests" y finalmente "silence run".
- Implemente la solución a los ejercicios en los archivos **sql/solucion.sql** (para código sql) y **tests/solucion.http** (para peticiones http). Estos ejercicios están basados en unos requisitos adicionales que se suministran a continuación.
- Incluya capturas de pantalla HeidiSQL/vsCode que muestre el resultado de ejecutar el ejercicio al final de este documento.
- Debe subir a la actividad de EV este documento con las capturas de pantallas, y los archivos sql/solucion.sql y tests/solucion.http. Antes de subir la actividad, avise a un profesor para que valide la misma.
- Es necesario completar un nivel para que se valoren los siguientes niveles.

Catálogo de requisitos

RI-1-01 Requisito de información	
Como:	Profesor de la asignatura
Quiero:	Tener disponible información sobre Multas : Una multa es una penalización a un usuario por alguna causa. Sus atributos son: el usuario multado, la causa de la multa, un entero representando el nivel de la multa, la fecha de la multa, y si es moderada o grave. Todos los atributos son obligatorios.
Para:	Evaluar al estudiante
RN-1-01 Primera regla de negocio	
Como:	Profesor de la asignatura
Quiero:	Un usuario no puede tener varias multas el mismo día.
Para:	Evaluar al estudiante
RN-1-02 Segunda regla de negocio	
Como:	Profesor de la asignatura
Quiero:	El nivel debe estar entre 0 y 10 (inclusive).
Para:	Evaluar al estudiante
RN-1-03 Tercera regla de negocio	
Como:	Profesor de la asignatura
Quiero:	Una multa grave no puede tener un nivel inferior a 3.
Para:	Evaluar al estudiante

PA- 1-01 Inserción de nuevas entidades	
Como:	Profesor de la asignatura
Quiero:	<ul style="list-style-type: none">✓ Insertar Multa con causa "Multa 1" del usuario con ID=2, de nivel 3, moderada, del 05/08/2020.✓ Insertar Multa con causa "Multa 2" del usuario con ID=2, de nivel 3, moderada, del 06/09/1997.✓ Multa con causa "Multa 3" del usuario con ID=1, de nivel 1 moderada, del 04/06/2016.✗ Insertar Multa con causa "Multa 4" del usuario con ID=1, de nivel 1, grave, del 02/01/2090. (RN-1-03)✗ Insertar Multa con causa "Multa 5" del usuario con ID=20, de nivel 10, grave, del 04/02/2020. (referencia no existente)✗ Insertar Multa con causa "Multa 6" del usuario con ID=2, de nivel 1, moderada, del 05/08/2020. (RN-1-01)
Para:	Evaluar al estudiante.

Nivel C. Máximo 7 puntos

1.(2)- Implemente los requisitos proporcionados (RI/RN) en MariaDB.

2.(1)- Cree un procedimiento almacenado pInsert(<atributos obligatorios>) para llevar a cabo la inserción de objetos del RI suministrado.

Ejecute llamadas a pInsert para implementar las pruebas de aceptación. **Incluya captura de pantalla con los datos de la tabla después de las pruebas.**

Use Delete para eliminar el elemento con ID=1. **Incluya captura de pantalla con los datos de la tabla después de la llamada.**

Use Update para actualizar los elementos con IDs 2 y 3 para que tengan un nivel de 7. **Incluya captura de pantalla con los datos de la tabla después de las llamadas.**

3.(0.5)- Escriba y realice peticiones HTTP para obtener el listado de usuarios de la base de datos.

4.(1)- Implemente una consulta SQL que devuelva los usuarios con al menos una foto:

idUser
4
3
2
1

5.(1.5)- Cree un disparador llamado tCorrectLevels que, al actualizarse una Multa, si el nivel fuera menor a 0, lo establezca como 0 en su lugar. Ejecute Update con valor -5 para comprobar que funciona el disparador. Tenga en cuenta que la ejecución de este procedimiento debe hacer saltar al disparador y que el valor final actualizado sea correcto. **Incluya captura de pantalla con los datos de la tabla después de ejecutar la llamada.**

6.(1)- Cree una función que, a partir del identificador de una Multa, devuelva nivel*10. Use la función para obtener el resultado a partir del elemento con ID=2. Incluya capturas de pantallas.

Nivel B. Máximo 9 puntos

7.(1)- Implemente una consulta SQL que devuelva los usuarios mayores de 30 años junto con la cantidad de fotos que tienen.

 idUser	name	 email	passwd	age	numPhotos
1	David Ruiz	druiz@us.es	pbkdf2:sha256:150000\$rFgsC...	45	3
3	Carlos Arévalo	carevalo@us.es	pbkdf2:sha256:150000\$HtZ8o...	55	2
4	Alfonso Márquez	amarquez@us.es	pbkdf2:sha256:150000\$R1c5...	35	1

8.(1)- Implemente las pruebas de aceptación como peticiones POST HTTP. **Recuerde limpiar los datos de la tabla insertados los apartados anteriores.** Las peticiones POST se pueden realizar con o sin autenticación. Incluya capturas de pantallas.

Nivel A. Máximo 10 puntos

9.(0.5)- Implemente una consulta SQL que devuelva un listado con la cantidad de fotos de cada categoría con una descripción que empiece por 'P'.

 idCategory	description	numberPhotos
2	Paisajes	2
3	Personas	4
7	Playa	2

10.(0.5)- Cree un procedimiento **pAddTwoPenalties(...)** que, dentro de una transacción, inserte dos Multas a partir de los datos suministrados de los dos.

Realice dos llamadas: una que inserte dos correctamente, y una en la que el segundo rompa alguna restricción y aborte la transacción. Incluya capturas de pantallas

Capturas de pantalla

Inserte aquí las capturas de pantalla COMPLETA que muestre la ejecución de los ejercicios. Para realizar una captura de pantalla puede usar la tecla “impr pant” del teclado y pulsar “ctrl+v” para pegar la captura de pantalla.

Ejercicio 1

```

5
6 -- A continuación debe escribir el código SQL que da solución a las cuestiones planteadas
7 -- puede añadir todos los comentarios que estime oportuno
8
9 DROP TABLE IF EXISTS Multas;
10 -- Ejercicio 1
11 CREATE TABLE Multas(
12     idMulta INT NOT NULL AUTO_INCREMENT,
13     idUser INT NOT NULL,
14     causa VARCHAR(64) NOT NULL,
15     nivel INT NOT NULL,
16     fecha DATE NOT NULL,
17     tipo ENUM('Moderada', 'Grave') NOT NULL,
18     PRIMARY KEY (idMulta),
19     FOREIGN KEY (idUser) REFERENCES users (idUser),
20     UNIQUE(idMulta, idUser, fecha),
21     CONSTRAINT RN_1_02 CHECK (nivel >= 0 AND nivel <= 10)
22 );
23
24 DELIMITER //
25 CREATE OR REPLACE TRIGGER RN_1_03
26 BEFORE INSERT ON Multas
27 FOR EACH ROW
28 BEGIN
29     IF (NEW.tipo='Grave' AND NEW.nivel >= 3)
30     THEN SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Una multa grave no puede tener un nivel inferior a 3.';
31     END IF;
32 END //
33 DELIMITER ;

```

Columnas: + Agregar ✖ Borrar ▲ Subir ▼ Bajar											
#	Nombre	Tipo de datos	Longitud/Co...	Sin signo	Permitir...	Relle...	Predeterminado	Comentario	Collation	Expresión	Virtualidad
1	idMulta	INT	11	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	AUTO_INCREME...				
2	idUser	INT	11	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Sin valor predeter...				
3	causa	VARCHAR	64	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Sin valor predeter...		latin1_swedish_ci		
4	nivel	INT	11	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Sin valor predeter...				
5	fecha	DATE		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Sin valor predeter...				
6	tipo	ENUM	'Moderada',...	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL		latin1_swedish_ci		

Ejercicio 2

```

34
35 -- Ejercicio 2
36
37 DELIMITER //
38 CREATE OR REPLACE PROCEDURE pInsert(i INT, c VARCHAR(64), n INT, f DATE, t ENUM('Moderada', 'Grave'))
39 BEGIN
40     INSERT INTO multas(idUser, causa, nivel, fecha, tipo) VALUES (i,c,n,f,t);
41 END //
42 DELIMITER ;
43
44 CALL pInsert(2,'Multa 1', 3,'2020-08-05','Moderada');
45 CALL pInsert(2,'Multa 2', 3,'1997-09-06','Moderada');
46 CALL pInsert(1,'Multa 3', 1,'2016-06-04','Moderada');
47 -- CALL pInsert(20,'Multa 6', 3,'1997-09-06','Moderada');
48
49 -- Insertar Multa con causa "Multa 6" del usuario con ID=2, de nivel 1, moderada, del 05/08/2020. (RN-1-01)
50
51
52 DELETE FROM multas WHERE multas.idMulta=1;
53
54
55 UPDATE multas SET nivel = 7 WHERE idMulta=2;
56 UPDATE multas SET nivel = 7 WHERE idMulta=3;
57

```

herramientas Ir a Ayuda

Host: 127.0.0.1 Base de datos: modeloexamen Tabla: multas Datos Consulta* soluciones

modeloexamen.multas: 3 filas en total (aproximadamente)

idMulta	idUser	causa	nivel	fecha	tipo
1	2	Multa 1	3	2020-08-05	Moderada
2	2	Multa 2	3	1997-09-06	Moderada
3	1	Multa 3	1	2016-06-04	Moderada

11.3.0.6295

ntas Ir a Ayuda

Host: 127.0.0.1 Base de datos: modeloexamen Tabla: multas Datos Consulta*

modeloexamen.multas: 2 filas en total (aproximadamente)

idMulta	idUser	causa	nivel	fecha	tipo
2	2	Multa 2	3	1997-09-06	Moderada
3	1	Multa 3	1	2016-06-04	Moderada

R:\modeloexamen\multas\ - HeidiSQL 11.3.0.6295

Editar Buscar Consulta Herramientas Ir a Ayuda

ases de c	Filtro de tablas	Host: 127.0.0.1	Base de datos: modeloexamen	Tabla: multas	Datos	Consulta*	solucion.sql*
modeloexamen.multas: 2 filas en total (aproximadamente)							
idMulta	idUser	causa	nivel	fecha	tipo		
2	2	Multa 2	7	1997-09-06	Moderada		
3	1	Multa 3	7	2016-06-04	Moderada		

Ejercicio 3

Response(18ms) - examen-lab-iissi - Visual Studio Code

settings.py M

solucion.http M

```
tests > solucion.http > GET /api/v1/users
1  ### Respuestas al examen de laboratorio.
2  ### Nombre del alumno:
3  ### Grupo al que pertenece:
4
5  ### A continuación debe escribir las pet
6  ### solución a las cuestiones planteadas
7  ### los comentarios que estime oportuno
8
9  1 reference
10 @BASE = http://127.0.0.1:8081/api/v1
11 GET {{BASE}}/users
12
```

Response(18ms)

```
13  "name": "David Ruiz",
14  "passwd": "pbkdf2:sha256:150000$nrFgsCpnI$884d47b3e5848d88bbf43eb22e1152af7f885e10
    f6f3f2ec31fd4fe184f20be3"
15  },
16  {
17    "age": 28,
18    "email": "dayala1@us.es",
19    "idUser": 2,
20    "name": "Daniel Ayala",
21    "passwd": "pbkdf2:sha256:150000$F0d4gXg8$e8d2e6ec3e15dae111cc275afffebcdbe7ecd4f34
    faad0d60abefca63ba4c2204"
22  },
23  {
24    "age": 55,
25    "email": "carevalo@us.es",
26    "idUser": 3,
27    "name": "Carlos Ar\00e9valo",
28    "passwd": "pbkdf2:sha256:150000$HtZ8oG60$3ca0720ca83d0ec35e4ec1e579d1b0fec94c2857
    d64dd32f08dc527ba97eb383"
29  },
30  {
31    "age": 35,
32    "email": "amarquez@us.es",
33    "idUser": 4,
34    "name": "Alfonso M\00e1rquez",
35    "passwd": "pbkdf2:sha256:150000$R1c5MR7y$e2dc906159bec8c79aa0b5b3ac84236b3b2513cb
    c70741be87592a05999e545c"
36  }
37  ]
```

Ejercicio 4

```
0
1  -- EJERCICIO 4
2
3  SELECT users.idUser FROM users
4  JOIN photos ON (users.idUser = photos.idUser)
5  GROUP BY users.idUser
6  HAVING COUNT(photos.idPhoto > 1);
7
8
9
```

Ejercicio 5

```
63
64  -- Ejercicio 5
65  DELIMITER //
66  CREATE OR REPLACE TRIGGER tCorrectLevels
67  BEFORE UPDATE ON Multas
68  FOR EACH ROW
69  BEGIN
70      IF(NEW.NIVEL < 0) THEN SET NEW.NIVEL = 0;
71      END IF;
72  END //
73  DELIMITER //
74
75  UPDATE multas SET NIVEL = -5 WHERE idMultas=2;
76
```

.0.6295

Ir a Ayuda

Host: 127.0.0.1 Base de datos: modeloexamen Tabla: multas Datos Consulta* solucion.sql*						
modeloexamen.multas: 2 filas en total (aproximadamente)						
idMultas	idUser	causa	nivel	fecha	tipo	
2	2	Multa 2	0	1997-09-06	Moderada	
3	1	Multa 3	1	2016-06-04	Moderada	

Ejercicio 6

```
76 -- Ejercicio 6
77 DELIMITER //
78 CREATE OR REPLACE FUNCTION fNivel10(id INT) RETURNS INT
79 BEGIN
80     DECLARE niveles INT;
81     SET niveles = (SELECT nivel FROM multas WHERE multas.idMulta=id);
82     RETURN niveles*10;
83 END //
84 DELIMITER ;
85 SELECT fNivel10(2);
86
87
88
89
```

users (4r x 1c) Resultado #2 (1r x 1c)

fNivel10(2)
30

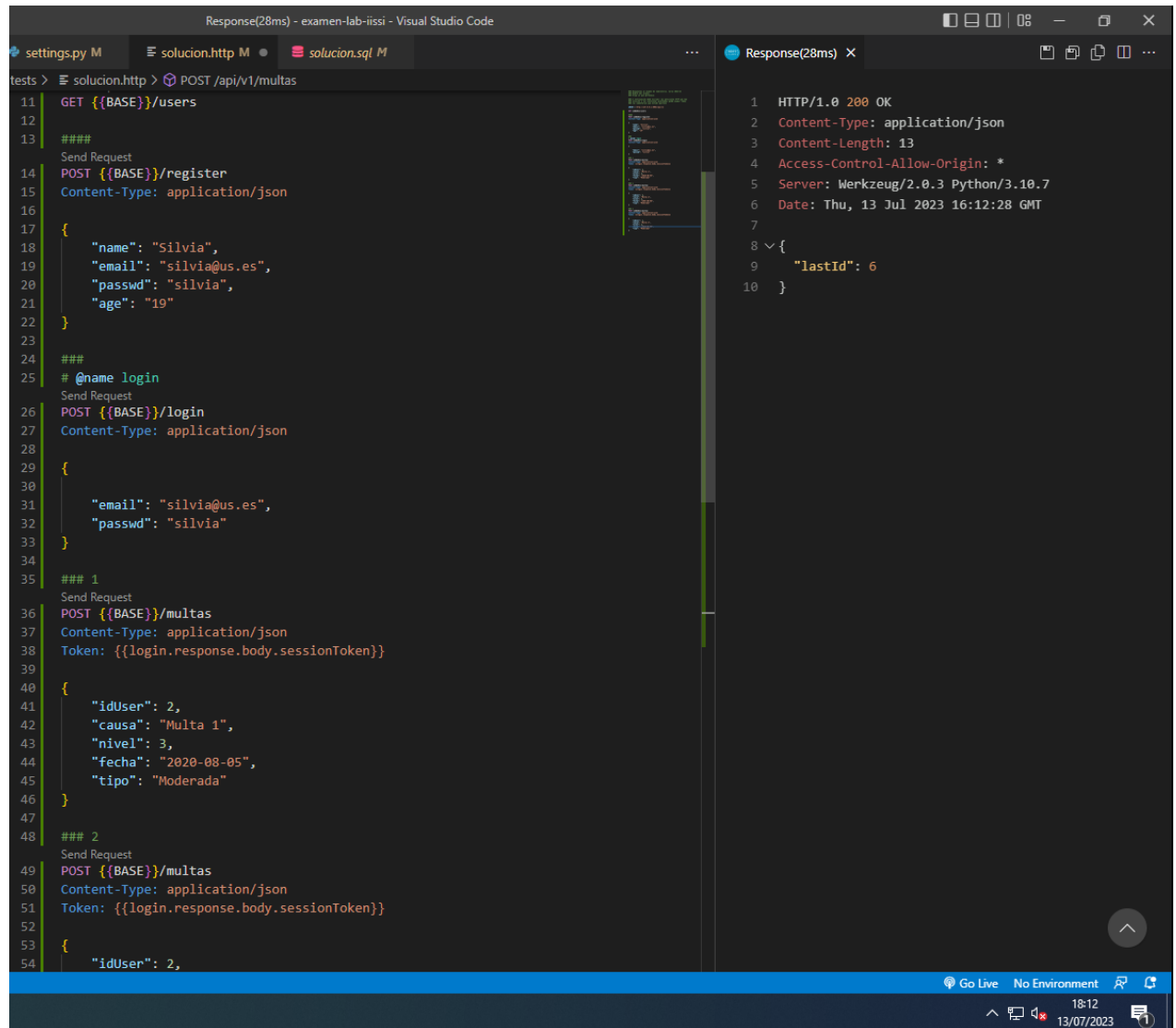
Ejercicio 7

```
88 -- Ejercicio 7
89
90 SELECT users.idUser, users.name, users.email, users.passwd, users.age, COUNT(photos.idPhoto) AS numPhotos FROM users
91 JOIN photos ON (users.idUser = photos.idUser)
92 GROUP BY users.idUser
93 HAVING age>30;
94
95
```

users (4r x 1c) Resultado #2 (1r x 1c) users (3r x 6c)

iUser	name	email	passwd	age	numPhotos
1	David Ruiz	druiz@us.es	pbkdf2:sha256:150000\$rFgsCpnl\$884d47b3e5848d88...	45	3
3	Carlos Arévalo	carevalo@us.es	pbkdf2:sha256:150000\$HtZ8oG6O\$3ca0720ca83d0ec...	55	2
4	Alfonso Márquez	amarquez@us.es	pbkdf2:sha256:150000\$R1c5MR7y\$e2dc906159bec8c...	35	1

Ejercicio 8



The image shows a Visual Studio Code editor with a REST client file named 'solucion.http'. The file contains three HTTP requests: a GET request for '/users', a POST request for '/register', and a POST request for '/multas'. The first two requests are commented out. The third request is active and has a corresponding response in the 'Response(28ms)' panel. The response is an HTTP 200 OK with a JSON body containing 'lastId': 6.

```
11 GET {{BASE}}/users
12
13 #####
14 Send Request
15 POST {{BASE}}/register
16 Content-Type: application/json
17 {
18     "name": "Silvia",
19     "email": "silvia@us.es",
20     "passwd": "silvia",
21     "age": "19"
22 }
23
24 ###
25 # @name login
26 Send Request
27 POST {{BASE}}/login
28 Content-Type: application/json
29 {
30     "email": "silvia@us.es",
31     "passwd": "silvia"
32 }
33
34 ### 1
35 Send Request
36 POST {{BASE}}/multas
37 Content-Type: application/json
38 Token: {{login.response.body.sessionToken}}
39
40 {
41     "idUser": 2,
42     "causa": "Multa 1",
43     "nivel": 3,
44     "fecha": "2020-08-05",
45     "tipo": "Moderada"
46 }
47
48 ### 2
49 Send Request
50 POST {{BASE}}/multas
51 Content-Type: application/json
52 Token: {{login.response.body.sessionToken}}
53
54 {
55     "idUser": 2,
```

Response(28ms)

```
1 HTTP/1.0 200 OK
2 Content-Type: application/json
3 Content-Length: 13
4 Access-Control-Allow-Origin: *
5 Server: Werkzeug/2.0.3 Python/3.10.7
6 Date: Thu, 13 Jul 2023 16:12:28 GMT
7
8 {
9     "lastId": 6
10 }
```

Response(28ms) - examen-lab-iissi - Visual Studio Code

```
settings.py M  solucion.http M  solucion.sql M  ...  Response(28ms) X
```

```
sts > E solucion.http > POST /api/v1/multas
Content-Type: application/json
Token: {{login.response.body.sessionToken}}

{
  "idUser": 2,
  "causa": "Multa 1",
  "nivel": 3,
  "fecha": "2020-08-05",
  "tipo": "Moderada"
}

### 2
Send Request
POST {{BASE}}/multas
Content-Type: application/json
Token: {{login.response.body.sessionToken}}

{
  "idUser": 2,
  "causa": "Multa 2",
  "nivel": 3,
  "fecha": "1997-09-06",
  "tipo": "Moderada"
}

### 3
Send Request
POST {{BASE}}/multas
Content-Type: application/json
Token: {{login.response.body.sessionToken}}

{
  "idUser": 1,
  "causa": "Multa 3",
  "nivel": 1,
  "fecha": "2016-04-06",
  "tipo": "Moderada"
}
```

```
1 HTTP/1.0 200 OK
2 Content-Type: application/json
3 Content-Length: 13
4 Access-Control-Allow-Origin: *
5 Server: Werkzeug/2.0.3 Python/3.10.7
6 Date: Thu, 13 Jul 2023 16:12:28 GMT
7
8 {
9   "lastId": 6
10 }
```

Host: 127.0.0.1 Base de datos: modeloexamen Tabla: multas Datos Consulta* solucion.sql

modeloexamen.multas: 3 filas en total (aproximadamente)

idMulta	idUser	causa	nivel	fecha	tipo
4	2	Multa 1	3	2020-08-05	Moderada
5	2	Multa 2	3	1997-09-06	Moderada
6	1	Multa 3	1	2016-04-06	Moderada

Ejercicio 9

```

89     HAVING age>30;
90
91 -- Ejercicio 9
92 SELECT categories.idCategory, categories.description, COUNT(photos.idPhoto) AS numPhotos
93 FROM categories NATURAL JOIN categoriesphotos NATURAL JOIN photos
94 GROUP BY categories.idCategory
95 HAVING description LIKE ('P%');
96
97 -- Ejercicio 10

```

users (4r x 1c) Resultado #2 (1r x 1c) users (3r x 6c) categories (3r x 3c)

idCategory	description	numPhotos
2	Paisajes	2
3	Personas	4
7	Playa	2

Ejercicio 10

```

103 -- Ejercicio 10
104 DELIMITER //
105 CREATE OR REPLACE PROCEDURE pAddTwoPenalties(i1 INT, c1 VARCHAR(64), n1 INT, f1 DATE, t1 ENUM('Moderada', 'Grave'),
106 i2 INT, c2 VARCHAR(64), n2 INT, f2 DATE, t2 ENUM('Moderada', 'Grave'))
107 BEGIN
108     START TRANSACTION;
109     tblock: BEGIN
110         DECLARE EXIT HANDLER FOR SQLEXCEPTION, SQLWARNING
111         BEGIN
112             ROLLBACK;
113             RESIGNAL;
114         END;
115         INSERT INTO multas (idUser, causa, nivel, fecha, tipo) VALUES (i1,c1,n1,f1,t1);
116         INSERT INTO multas (idUser, causa, nivel, fecha, tipo) VALUES (i2,c2,n2,f2,t2);
117     COMMIT;
118     END tblock;
119 END //
120 DELIMITER ;
121
122 CALL pAddTwoPenalties(2,'Multa 4', 3,'2021-08-05','Moderada',2,'Multa 5', 3,'2019-08-05','Moderada');
123 CALL pAddTwoPenalties(20,'Multa 4', 80,'2021-08-05','Moderada',2,'Multa 5', 13,'2019-08-05','Moderada');
124

```

modeloexamen.multas: 5 filas en total (aproximadamente)

idMulta	idUser	causa	nivel	fecha	tipo
4	2	Multa 1	3	2020-08-05	Moderada
5	2	Multa 2	3	1997-09-06	Moderada
6	1	Multa 3	1	2016-04-06	Moderada
7	2	Multa 4	3	2021-08-05	Moderada
8	2	Multa 5	3	2019-08-05	Moderada

```
-- Ejercicio 10
DELIMITER //
CREATE OR REPLACE PROCEDURE pAddTwoPenalties(i1 INT, c1 VARCHAR(64), n1 INT, f1 DATE, t1 ENUM('Moderada', 'Grave'),
i2 INT, c2 VARCHAR(64), n2 INT, f2 DATE, t2 ENUM('Moderada', 'Grave'))
BEGIN
  START TRANSACTION;
  tblock: BEGIN
    DECLARE EXIT HANDLER FOR SQLEXCEPTION, SQLWARNING
    BEGIN
      ROLLBACK;
      RESIGNAL;
    END;
    INSERT INTO multas (idUser, causa, nivel, fecha, tipo) VALUES (i1,c1,n1,f1,t1);
    INSERT INTO multas (idUser, causa, nivel, fecha, tipo) VALUES (i2,c2,n2,f2,t2);
  COMMIT;
END tblock;
END //
DELIMITER ;

CALL pAddTwoPenalties(2,'Multa 4', 3,'2021-08-05','Moderada',2,'Multa 5', 3,'2019-08-05','Moderada');
CALL pAddTwoPenalties(20,'Multa 4', 80,'2021-08-05','Moderada',20,'Multa 5', 80,'2019-08-05','Moderada');
```

