



# Act.01. UT05. Edición de datos

Bases de datos

Francisco José García Cutillas | 1FPGS\_DAM



## Índice

Ejercicio 1 .....	3
Ejercicio 2 .....	3
Ejercicio 3 .....	4
Ejercicio 4 .....	4
Ejercicio 5 .....	5
Ejercicio 6 .....	5
Ejercicio 7 .....	6
Ejercicio 8 .....	6
Ejercicio 9 .....	6
Ejercicio 10 .....	7
Ejercicio 11 .....	8
Ejercicio 12 .....	8
Ejercicio 13 .....	9
Ejercicio 14 .....	9
Ejercicio 15 .....	9
Ejercicio 16 .....	10
Ejercicio 17 .....	10
Ejercicio 18 .....	11
Ejercicio 19 .....	12
Ejercicio 20 .....	12
Ejercicio 21 .....	13
Ejercicio 22 .....	14
Ejercicio 23 .....	14
Ejercicio 24 .....	15
Ejercicio 25 .....	16
Ejercicio 26 .....	16
Ejercicio 27 .....	17
Ejercicio 28 .....	17
Ejercicio 29 .....	18
Ejercicio 30 .....	19
Ejercicio 31 .....	20

Utilizando la base de datos tienda (que se adjunta a la práctica) realiza las siguientes actividades.

**Nota:** Solo se confirmarán o se desecharán los cambios en aquel momento que se indique. Si algún apartado NO se puede realizar, es necesario hacer una explicación detallada del problema que hace que no se pueda efectuar el mismo (no bastará con poner el mensaje del error, sino que será necesario explicar el mismo).

**NOTA:** Es posible que algunos apartados se puedan/tengan que hacer con más de una operación/instrucción. Por ejemplo, en el caso de que tenga que insertar 4 provincias, se pueden hacer 4 instrucciones y añadir una provincia por instrucción.

## Ejercicio 1

Comprueba que tienes desactivada la confirmación automática. En Workbench mirar en Preferences SQL Editor - SQL Execution (set autocommit = 0; show variables like 'autocommit';)

```
4
5 • set autocommit = 0;
6 • show variables like 'autocommit';
```

	Variable_name	Value
▶	autocommit	OFF

## Ejercicio 2

Crea una nueva tabla que se llame provincia y que tendrá dos columnas código que será de tipo entero y nombre que tendrá una longitud máxima de 60.

```
9
10 -- Ejercicio 2
11 • create table provincia (
12     codigo int primary key,
13     nombre varchar (60));
14
```

## Ejercicio 3

En la tabla fabricantes añade una nueva columna en la que vamos a almacenar el código de la provincia. ¿Nulo o no nulo? ¿Por qué? Crea la clave ajena correspondiente de la forma que consideres.

```
16  -- Ejercicio 3
17 • alter table fabricantes
18     add codigo_provincia int;
19
20 • alter table fabricantes
21     add constraint fk_fabricantes_provincia foreign key (codigo_provincia) references provincia (codigo)
22     on delete restrict on update cascade;
23
```

Ahora mismo la columna "código provincia" la vamos a crear para que permita nulos, puesto que en la tabla "provincia" actualmente no hay ningún dato registrado. Además, si lo pusiéramos como no nulo, no podríamos definir la foreign key hacia la tabla "provincia".

## Ejercicio 4

Añadir un nuevo producto con código 11, nombre Altavoces de 70 euros del fabricante 2.

```
25  -- Ejercicio 4
26 • insert into articulos (codigo, nombre, precio, fabricante) values (11, 'Altavoces', 70, 2);
27
28
```

	Codigo	Nombre	Precio	Fabricante
▶	1	Tclado	100	3
	2	Disco Duro 300Gb	500	5
	3	Mouse	80	3
	4	Memoria USB	140	4
	5	Memoria RAM	290	1
	6	Disco Duro extraible 250Gb	650	5
	7	Memoria USB	279	1
	8	DVD Rom	450	2
	9	CD Rom	200	2
	10	Tarjeta de red	180	3
	11	Altavoces	70	2
*	NULL	NULL	NULL	NULL

## Ejercicio 5

Añadir un nuevo producto con código 12, nombre Auricular Bluetooth de 27 euros del fabricante 2 (utiliza un formato diferente para este apartado del empleado en el apartado anterior).

```
29 -- Ejercicio 5
30 • insert into articulos (codigo, nombre, precio, fabricante) values (12, 'Auricular Bluetooth', 27, 2);
31
--
```

	Codigo	Nombre	Precio	Fabricante
▶	1	Tedado	100	3
	2	Disco Duro 300Gb	500	5
	3	Mouse	80	3
	4	Memoria USB	140	4
	5	Memoria RAM	290	1
	6	Disco Duro extraible 250Gb	650	5
	7	Memoria USB	279	1
	8	DVD Rom	450	2
	9	CD Rom	200	2
	10	Tarjeta de red	180	3
	11	Altavoces	70	2
	12	Auricular Bluetooth	27	2
•	NULL	NULL	NULL	NULL

## Ejercicio 6

Crea una tabla similar a artículos y llámala articulos\_adata.

```
33 -- Ejercicio 6
34 • create table articulos_adata like articulos;
35
--
```

	Codigo	Nombre	Precio	Fabricante
•	NULL	NULL	NULL	NULL

## Ejercicio 7

Mete en la tabla `articulos_adata` todos los artículos que sean del fabricante `adata` (ojo, debes hacerlo en una única instrucción).

```

37  -- Ejercicio 7
38  •  insert into articulos_adata
39      select a.codigo, a.nombre, a.precio, a.fabricante
40      from fabricantes f inner join articulos a
41      on f.codigo = a.fabricante
42      where f.nombre = 'Adata';
43

```

Result Grid				
	Codigo	Nombre	Precio	Fabricante
▶	8	DVD Rom	450	2
	9	CD Rom	200	2
	11	Altavoces	70	2
	12	Auricular Bluetooth	27	2
✱	NULL	NULL	NULL	NULL

## Ejercicio 8

Confirma lo realizado.

```

45  -- Ejercicio 8
46  •  commit;
47

```

✓	83	13:44:39	commit	0 row(s) affected	0.000 sec
---	----	----------	--------	-------------------	-----------

## Ejercicio 9

En la siguiente dirección tienes los códigos de provincias (columna CPRO) así como nombre de Provincia que suelen usarse para en bases de datos [https://www.ine.es/daco/daco42/codmun/cod\\_ccaa\\_provincia.htm](https://www.ine.es/daco/daco42/codmun/cod_ccaa_provincia.htm).  
Añade las provincias Burgos, Palencia, Toledo y Murcia.

```

49  -- Ejercicio 9
50  •  insert into provincia (codigo, nombre) values (09, 'Burgos'), (34, 'Palencia'), (45, 'Toledo'), (30, 'Murcia');
51

```

Result Grid			Filter Rows:
	codigo	nombre	
▶	9	Burgos	
	30	Murcia	
	34	Palencia	
	45	Toledo	
✱	NULL	NULL	

## Ejercicio 10

Los Fabricantes que tienen un código inferior a 3 van a tener asignada la provincia de Murcia y el resto van a tener asignada la provincia con código 08.

```

53      -- Ejercicio 10
54 •    update fabricantes
55      set codigo_provincia = (select codigo
56                               from provincia
57                               where nombre = 'Murcia')
58      where codigo < 3;
59

```

Result Grid				Filter Rows:
	Codigo	Nombre	codigo_provincia	
▶	1	Kingston	30	
	2	Adata	30	
	3	Logitech	NULL	
	4	Lexar	NULL	
	5	Seagate	NULL	
✱	NULL	NULL	NULL	

```

--
60 •    update fabricantes
61      set codigo_provincia = 08
62      where codigo >= 3;
63

```

✖	100	14:20:01	update fabricantes set codigo_provincia = 08 where codigo >= 3	Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails ('ti..
---	-----	----------	--	---

En este caso nos da error debido a que estamos intentando asignar un código de provincia de una provincia, valga la redundancia, que no hay en la base de datos. Esto es así debido a que tenemos creada una foreign key que une la tabla “fabricantes” con la de “provincia” con sus campos “código\_provincia” y “código” respectivamente.

## Ejercicio 11

Confirma lo realizado.

```
65      -- Ejercicio 11
66 •    commit;
67
```

✓ 101 14:23:46 commit 0 row(s) affected

## Ejercicio 12

Actualiza para los fabricantes que no tienen provincia (si es que hubiera alguno) con la provincia que tenga un código más grande.

```
69      -- Ejercicio 12
70 •    update fabricantes
71      set codigo_provincia = (select max(codigo)
72                               from provincia)
73      where codigo_provincia is null;
74
```

Result Grid			
	Codigo	Nombre	codigo_provincia
▶	1	Kingston	30
	2	Adata	30
	3	Logitech	45
	4	Lexar	45
	5	Seagate	45
•	NULL	NULL	NULL



## Ejercicio 13

Cambia la columna provincia en la tabla Fabricantes para que deba tener obligatoriamente un código de provincia asignado. Cambia también si fuera necesario la foreign key de Fabricantes a Provincia para que el borrado sea restringido y la actualización en cascada.

```
76 -- Ejercicio 13
77 • alter table fabricantes
78     modify column codigo_provincia int not null;
79
80 • show fields from fabricantes;
```

R1

Result Grid						
	Field	Type	Null	Key	Default	Extra
▶	Codigo	int	NO	PRI	NULL	
	Nombre	varchar(100)	YES		NULL	
	codigo_provincia	int	NO	MUL	NULL	

## Ejercicio 14

Confirma lo realizado.

```
83 -- Ejercicio 14
84 • commit;
85
```

✓	112	12:06:54	commit	0 row(s) affected
---	-----	----------	--------	-------------------

## Ejercicio 15

Añadir un nuevo producto con código 13, nombre Micrófono de 12 euros del fabricante 13

```
--
87 -- Ejercicio 15
88 • insert into articulos (codigo, nombre, precio, fabricante) values (13, 'Microfono', 12, 13);
89
90 Error Code: 1452. Cannot add or update a child row: a foreign key constraint
91 fails (`tienda`.`articulos`, CONSTRAINT `articulos_ibfk_1` FOREIGN KEY (`Fabricante`)
92 REFERENCES `fabricantes` (`Codigo`) ON DELETE RESTRICT ON UPDATE CASCADE)
93
--
```

En este caso nos da error porque el campo “fabricante” de la tabla “articulos” es una foreign key del campo “código” de la tabla “fabricantes”, en la cual no hay ningún fabricante con el código 13.

## Ejercicio 16

**Cambiar el nombre del producto 8 a Impresora Láser.**

```

97      -- Ejercicio 16
98 •    update articulos
99        set nombre = 'Impresora laser'
100      where codigo = 8;
101

```

	Codigo	Nombre	Precio	Fabricante
▶	1	Tedado	100	3
	2	Disco Duro 300Gb	500	5
	3	Mouse	80	3
	4	Memoria USB	140	4
	5	Memoria RAM	290	1
	6	Disco Duro extraible 250Gb	650	5
	7	Memoria USB	279	1
	8	Impresora laser	450	2
	9	CD Rom	200	2
	10	Tarjeta de red	180	3
	11	Altavoces	70	2
	12	Auricular Bluetooth	27	2
*	NULL	NULL	NULL	NULL

## Ejercicio 17

**Incrementar en un 10% el precio de todos los productos**

```

103      -- Ejercicio 17
104 •    update articulos
105        set precio = precio * 1.1
106      where codigo > 0;

```

	Codigo	Nombre	Precio	Fabricante
1	Teclado		110	3
2	Disco Duro 300Gb		550	5
3	Mouse		88	3
4	Memoria USB		154	4
5	Memoria RAM		319	1
6	Disco Duro extraible 250Gb		715	5
7	Memoria USB		307	1
8	Impresora laser		495	2
9	CD Rom		220	2
10	Tarjeta de red		198	3
11	Altavoces		77	2
12	Auricular Bluetooth		30	2
*	NULL	NULL	NULL	NULL

En este caso hemos tenido que hacer una pequeña trampa para engañar al “safe update mode”, puesto que lo tenemos activado. Por ello no se pueden hacer actualizaciones sin usar el where. En nuestro caso hemos puesto como condición que el código del artículo sea mayor que 0, condición que cumplen todos los campos. También existiría la opción de desactivar el “safe update mode” con la instrucción “set sql\_safe\_updates = 0”.

Si no desactivamos el “safe update mode” o “engañamos” a mysql con un “where” que cumplan todas las filas, no se podrían actualizar todos los datos con una sola instrucción.

## Ejercicio 18

**Comprueba en las tablas si se han producido los cambios. Después aborta las operaciones y comprueba de nuevo el estado de las tablas. ¿Qué cambios ha producido esta operación?**

```
109      -- Ejercicio 18
```

```
110 •   rollback;
```

```
111
```

	Codigo	Nombre	Precio	Fabricante
1	Teclado		100	3
2	Disco Duro 300Gb		500	5
3	Mouse		80	3
4	Memoria USB		140	4
5	Memoria RAM		290	1
6	Disco Duro extraible 250Gb		650	5
7	Memoria USB		279	1
8	DVD Rom		450	2
9	CD Rom		200	2
10	Tarjeta de red		180	3
11	Altavoces		70	2
12	Auricular Bluetooth		27	2
*	NULL	NULL	NULL	NULL

Al realizar un “Rollback”, se han deshecho los cambios realizados en el ejercicio 16 y 17, puesto que en el ejercicio 15 a dar el error no realiza ningún cambio. Esto es así debido a que el último “Commit” se realizó en el ejercicio 14. Por lo tanto, se deshacen los cambios realizados en las tablas posteriores a dicho “Commit”.

## Ejercicio 19

**Aplicar un descuento de 8 euros a todos los productos del fabricante Adata.**

```

112  -- Ejercicio 19
113  •  update  articulos
114      set precio = precio - 8
115      where fabricante = (select codigo
116                          from fabricantes
117                          where nombre = 'Adata');
118

```

	Codigo	Nombre	Precio	Fabricante
▶	1	Tclado	100	3
	2	Disco Duro 300Gb	500	5
	3	Mouse	80	3
	4	Memoria USB	140	4
	5	Memoria RAM	290	1
	6	Disco Duro extraible 250Gb	650	5
	7	Memoria USB	279	1
	8	DVD Rom	442	2
	9	CD Rom	192	2
	10	Tarjeta de red	180	3
	11	Altavoces	62	2
	12	Auricular Bluetooth	19	2
*	NULL	NULL	NULL	NULL

## Ejercicio 20

**Añadir un nuevo fabricante con código 6 y nombre del fabricante Genius**

```

120  -- Ejercicio 20
121  •  insert into fabricantes (codigo, nombre) values (6, 'Genius');
122

```

✖ 184 18:11:41 insert into fabricantes (codigo, nombre) values (6, 'Genius') Error Code: 1364. Field 'codigo\_provincia' doesn't have a default value

En este caso, no se podría añadir un fabricante sin especificar el código de provincia, puesto que éste es una clave ajena a la tabla provincia, en la cual ese campo a su vez es clave primaria y por lo tanto no puede tener un valor nulo.

## Ejercicio 21

**Actualizar el precio del artículo 4 para que sea igual al precio más alto de la tabla artículos**

```

128  -- Ejercicio 21
129  •  update  artículos
130      set  precio = (select  precio_max
131                      from  (select  max(precio)  precio_max
132                              from  artículos) a)
133      where  codigo = 4;

```

Result Grid				
	Codigo	Nombre	Precio	Fabricante
▶	1	Tedado	100	3
	2	Disco Duro 300Gb	500	5
	3	Mouse	80	3
	4	Memoria USB	650	4
	5	Memoria RAM	290	1
	6	Disco Duro extraible 250Gb	650	5
	7	Memoria USB	279	1
	8	DVD Rom	442	2
	9	CD Rom	192	2
	10	Tarjeta de red	180	3
	11	Altavoces	62	2
	12	Auricular Bluetooth	19	2
*	NULL	NULL	NULL	NULL

En este caso tenemos que hacer una subconsulta dentro de la subconsulta, porque no se puede actualizar un campo de la misma tabla a la que hacemos la subconsulta. Al realizarlo de esta manera, lo camuflamos con la tabla que hemos llamado “a”. Así MySQL realiza la subconsulta y la nueva tabla ya no sería la misma que la de “artículos”.

## Ejercicio 22

**Actualizar el nombre y precio del artículo 2 a Portátil 300 GB, precio 450.**

```

136      -- Ejercicio 22
137 •    update articulos
138         set nombre = 'Portatil 300 GB', precio = 450
139         where codigo = 2;
140

```

	Codigo	Nombre	Precio	Fabricante
►	1	Tclado	100	3
	2	Portatil 300 GB	450	5
	3	Mouse	80	3
	4	Memoria USB	650	4
	5	Memoria RAM	290	1
	6	Disco Duro extraible 250Gb	650	5
	7	Memoria USB	279	1
	8	DVD Rom	442	2
	9	CD Rom	192	2
	10	Tarjeta de red	180	3
	11	Altavoces	62	2
	12	Auricular Bluetooth	19	2
▲	NULL	NULL	NULL	NULL

## Ejercicio 23

**Borrar de la tabla artículos aquellos cuyo precio sea menor que 150.**

```

142      -- Ejercicio 23
143 •    delete from articulos
144         where precio < 150
145         and codigo > 0;
146

```

En este caso hemos tenido que hacer lo mismo que en el ejercicio 17 debido al “safe update mode”

Result Grid				
		Filter Rows:		
	Codigo	Nombre	Precio	Fabricante
▶	2	Portatil 300 GB	450	5
	4	Memoria USB	650	4
	5	Memoria RAM	290	1
	6	Disco Duro extraible 250Gb	650	5
	7	Memoria USB	279	1
	8	DVD Rom	442	2
	9	CD Rom	192	2
	10	Tarjeta de red	180	3
*	NULL	NULL	NULL	NULL

## Ejercicio 24

**Borrar de la tabla fabricantes aquellos para los que no tenemos ningún artículo registrado.**

```

148  -- Ejercicio 24
149  •  set sql_safe_updates = 0;
150
151  •  delete from fabricantes
152      where codigo in (select f.cod codigo
153                      from (select f.codigo cod
154                          from articulos a right join fabricantes f
155                              on a.fabricante = f.codigo
156                              where a.fabricante is null) f);
157
158  •  set sql_safe_updates = 1;
159

```

	Codigo	Nombre	codigo_provincia
▶	1	Kingston	30
	2	Adata	30
	3	Logitech	45
	4	Lexar	45
	5	Seagate	45
*	NULL	NULL	NULL

En este caso, en lugar de hacer la trampa de los anteriores ejercicios, hemos deshabilitado el “sql\_safe\_updates”, el cual hemos vuelto a activar al final del ejercicio.

En nuestro caso no borraría ningún fabricante, puesto que todos los que tenemos hasta el momento tienen algún artículo en la base de datos.

## Ejercicio 25

Borrar de la tabla artículos aquellos cuyo nombre de fabricante comience por L.

```

161      -- Ejercicio 25
162 •    delete from articulos
163      where codigo in (select a.codigo
164                      from (select a.codigo codigo
165                          from articulos a inner join fabricantes f
166                          on a.fabricante = f.codigo
167                          where f.nombre like 'L%')a );
168

```

	Codigo	Nombre	Precio	Fabricante
▶	2	Portatil 300 GB	450	5
	5	Memoria RAM	290	1
	6	Disco Duro extraible 250Gb	650	5
	7	Memoria USB	279	1
	8	DVD Rom	442	2
	9	CD Rom	192	2
*	NULL	NULL	NULL	NULL

## Ejercicio 26

Borrar todos los datos de la tabla articulos\_adata

```

170      -- Ejercicio 26
171 •    set sql_safe_updates = 0;
172
173 •    delete from articulos_adata;
174
175 •    set sql_safe_updates = 1;
176

```

	Codigo	Nombre	Precio	Fabricante
*	NULL	NULL	NULL	NULL

Volvemos a repetir lo mismo que en el ejercicio 24. Si no desactivamos el “sql\_safe\_updates” no nos dejaría borrar todos los datos sin usar el “where”.



## Ejercicio 27

### Borrar todos los datos de la tabla fabricantes

Primero, hemos vuelto a desactivar el “sql\_safe\_updates”.

```

178      -- Ejercicio 27
179 •    set sql_safe_updates = 0;
180
181 •    delete from fabricantes;
182
183      /*
184      Error Code: 1451. Cannot delete or update a parent row: a foreign key constraint
185      fails (`tienda`.`articulos`, CONSTRAINT `articulos_ibfk_1` FOREIGN KEY (`Fabricante`)
186      REFERENCES `fabricantes` (`Codigo`) ON DELETE RESTRICT ON UPDATE CASCADE)
187      */

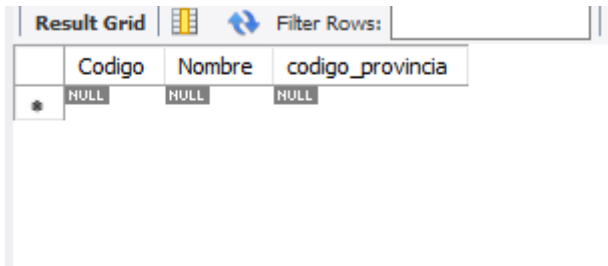
```

En nuestro caso no podremos eliminar los fabricantes ya que no nos lo permite, debido a que en la tabla “artículos” hay registros que hacen referencia a estos fabricantes y el borrado está restringido en la foreign key. Por ello, primeramente, tenemos que borrar los registros de la tabla “artículos” que hacen referencia a estos fabricantes.

```

189 •    delete from articulos;
190
191 •    delete from fabricantes;
192

```



	Codigo	Nombre	codigo_provincia
*	NULL	NULL	NULL

Tras ello, ya podemos borrar los registros de la tabla “fabricantes”.

## Ejercicio 28

### Borrar los DATOS que queden en la base de datos.

```

193
194      -- Ejercicio 28
195 •    delete from provincia;
196

```

	codigo	nombre
*	NULL	NULL

La única tabla que quedaba con datos era la de “provincia”

## Ejercicio 29

Comprueba en las tablas si se han producido los cambios. Después aborta las operaciones y comprueba de nuevo el estado de las tablas. ¿Qué cambios ha producido esta operación?

```
205 • select *
206     from articulos_adata;
207
```

	Codigo	Nombre	Precio	Fabricante
▶	8	DVD Rom	450	2
	9	CD Rom	200	2
	11	Altavoces	70	2
	12	Auricular Bluetooth	27	2
*	NULL	NULL	NULL	NULL


```
208 • select *
209     from articulos;
210
```

	Codigo	Nombre	Precio	Fabricante
▶	1	Teclado	100	3
	2	Disco Duro 300Gb	500	5
	3	Mouse	80	3
	4	Memoria USB	140	4
	5	Memoria RAM	290	1
	6	Disco Duro extraible 250Gb	650	5
	7	Memoria USB	279	1
	8	DVD Rom	450	2
	9	CD Rom	200	2
	10	Tarjeta de red	180	3
	11	Altavoces	70	2
	12	Auricular Bluetooth	27	2
*	NULL	NULL	NULL	NULL

```

211 • select *
212       from fabricantes;
213


```

Result Grid				
Filter Rows: <input type="text"/>				
Edit: 				
	Codigo	Nombre	Precio	Fabricante
▶	1	Tedado	100	3
	2	Disco Duro 300Gb	500	5
	3	Mouse	80	3
	4	Memoria USB	140	4
	5	Memoria RAM	290	1
	6	Disco Duro extraible 250Gb	650	5
	7	Memoria USB	279	1
	8	DVD Rom	450	2
	9	CD Rom	200	2
	10	Tarjeta de red	180	3
	11	Altavoces	70	2
	12	Auricular Bluetooth	27	2
*	NULL	NULL	NULL	NULL

```

214 • select *
215       from provincia;

```

Result Grid		
Filter Rows: <input type="text"/>		
Edit: 		
	codigo	nombre
▶	9	Burgos
	30	Murcia
	34	Palencia
	45	Toledo
*	NULL	NULL

El “rollback” ha deshecho todas las modificaciones que habíamos hecho posteriores al “commit” realizado en el ejercicio 14.

## Ejercicio 30

**Elimina la TABLA artículos y luego elimina la tabla fabricantes.**

```

201      -- Ejercicio 30
202 • drop table articulos;
203

```

268 20:18:57 select \* from articulos LIMIT 0, 1000 Error Code: 1146. Table 'tienda.articulos' doesn't exist

```
204 • drop table fabricantes;
205
```

❌ 270 20:20:16 select \* from fabricantes LIMIT 0, 1000 Error Code: 1146. Table 'tienda.fabricantes' doesn't exist

## Ejercicio 31

Comprueba en la base de datos si se han producido los cambios. Después aborta las operaciones y comprueba de nuevo el estado de las tablas. ¿Qué cambios ha producido esta operación?

```
204 -- Ejercicio 31
```

```
205 • show tables;
```

```
206
```

Result Grid		Filter Rows:
	Tables_in_tienda	
▶	articulos_adata	
	provincia	

```
207 • rollback;
```

```
208
```

```
209 • show tables;
```

```
210
```

Result Grid		Filter Rows:	Export
	Tables_in_tienda		
▶	articulos_adata		
	provincia		

En el caso de eliminar tablas, al realizar “rollback” no las vuelve a restaurar.