

AUTOR

Belén María Solís Valle

BASES DE DATOS DAW



Curso 2023/24

TAREA 03

Implantación de bases de datos relacionales

Descripción de la tarea

Caso práctico

Una empresa de gestión de clínicas sanitarias nos ha pedido que implementemos una base de datos para la gestión de las clínicas, los médicos que trabajan en ellas y los pacientes que acuden a operarse por los diversos médicos.

En la empresa para la que trabajas, estás formando equipo con José y María, dos compañeros que ya han realizado el diagrama Entidad-Relación y también el Modelo Relacional respectivamente. Tanto José como María ya han realizado los pasos previos, ahora es tu turno. Debes implementar la base de datos utilizando el lenguaje SQL para el Sistema Gestor de Bases de datos MySQL.

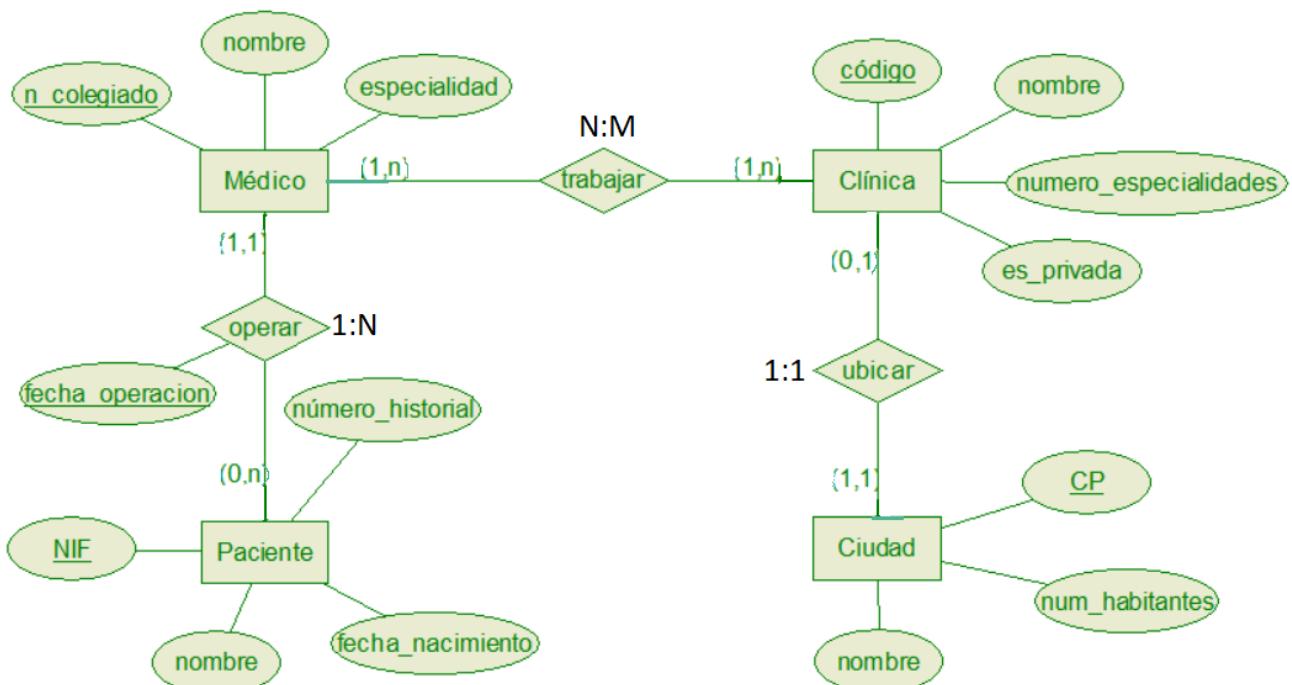
Es necesario que te apoyes tanto en el diagrama Entidad-Relación como en el Modelo Relacional y en las especificaciones que te han dejado anotadas para que realices un script SQL apropiado. Después de crear las tablas con todas las consideraciones oportunas, también tendrás que crear unas sentencias para modificar la estructura posterior o bien crear índices o vistas que son muy utilizadas también. Y por último deberás comprobar la gestión de usuarios creando uno de prueba y otorgando los permisos correspondientes.

¡ÁNIMO! ya estás utilizando un SGBD para implementar tu primera Base de Datos, en las unidades siguientes, seguirás aprendiendo más sobre el lenguaje SQL.

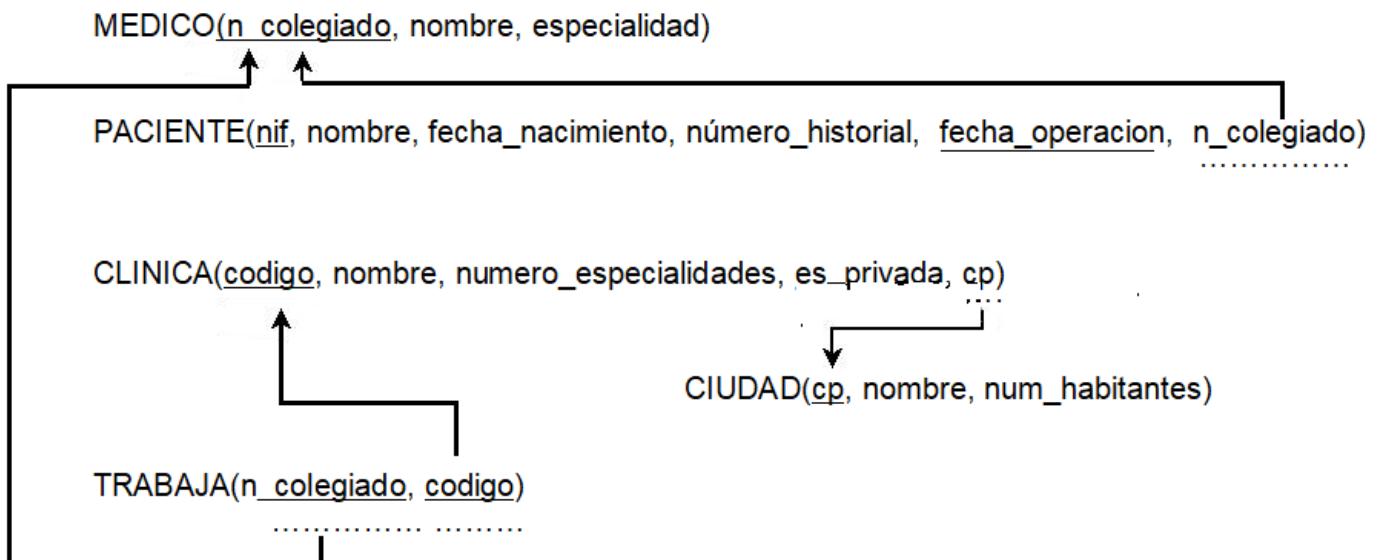
¿Qué te pedimos que hagas?

Después de haber realizado el análisis de todos los requerimientos que nos han pedido, se ha llegado al siguiente diagrama E-R y al Modelo Relacional correspondiente. Ahora, es el momento de implementar con MySQL dichas tablas y por tanto realizar un script con lenguaje DDL en SQL para que funcione perfectamente en dicho SGBD.

DIAGRAMA E-R



MODELO RELACIONAL



Partiendo de dichas tablas, debes realizar los siguientes apartados utilizando MySQL Workbench:

A) Escribe las sentencias de creación de tablas en el orden apropiado utilizando el lenguaje DDL de SQL para que funcionen correctamente en MySQL. Debes crear una sentencia SQL de creación de tabla para cada una de ellas teniendo en cuenta los siguientes subapartados:

- 1) Elegir el nombre, tipo de dato MySQL y el tamaño más adecuado para cada campo teniendo en cuenta los valores que éstos pueden almacenar.
- 2) Definir las claves primarias identificadas en cada una de las tablas.
- 3) Definir las claves ajenas identificadas en cada una de las tablas referenciando a los campos y tablas correspondientes y teniendo en cuenta las siguientes reglas de integridad referencial:
 - a) Cuando un valor de la tabla principal se elimine, se eliminarán también los valores a los que referencia, y además,
 - b) Cuando un valor de la tabla principal se actualice, se actualizarán los valores a los que referencia.
- 4) Incluir también en las sentencias de creación de tablas las siguientes restricciones y validaciones en los campos siguientes:
 - a) El numero_especialidades tendrá como valor predeterminado 1.
 - b) El campo cp almacena un valor que puede estar entre 00240 y 90007.
 - c) La fecha_operacion debe ser mayor o igual que la fecha_nacimiento.
 - d) El campo n_colegiado debe almacenar valores entre 1 y 999999999.
 - e) La fecha_nacimiento no puede quedar sin valor.
 - f) El número de historial es único por cada paciente, no se puede repetir.

Creamos la BBDD clínicas

The screenshot shows the MySQL Workbench interface. On the left, there is a sidebar for 'FORMACIÓN PROFESIONAL' with a profile picture of Belén Valle and some user information. The main window shows the 'MySQL Workbench' toolbar and the 'Navigator' panel which lists databases: 'clinicas', 'sakila', 'sys', and 'world'. The central pane displays a script titled 'Solis_Valle_Belen_Maria_BD_T...'. The script contains SQL code to drop and create the 'clinicas' database, and then switch to it. The 'Output' pane at the bottom shows the execution results:

```

1 ----- APARTADO A -----
2 /* Execute the selected portion of the script or everything, if there is no selection */
3 • drop database if exists clinicas;
4
5 /*Creamos la BBDD con nombre clinicas*/
6 • create database if not exists clinicas;
7
8 /*Ponemos la BBDD como por defecto*/
9 • use clinicas;
10
11 /*Creamos las tablas de la BBDD, primero hacemos
12 las que no tienen Clave foranea/foreign Key FK para que primero
13 se creen las Claves primarias/Primary Key PK que se
14 basaran despues las Foreign Key FK.*/
15
16 /*Creamos la tabla medico*/
17 • create table if not exists MEDICO (

```

Action Output				
#	Time	Action	Message	Duration / Fetch
1	03:52:38	create database if not exists clinicas	1 row(s) affected	0.015 sec

Usamos la base de datos clínicas

This screenshot is similar to the previous one but shows the 'use clinicas;' command being executed. The 'Output' pane now includes the second row of results:

Action Output				
#	Time	Action	Message	Duration / Fetch
1	03:52:38	create database if not exists clinicas	1 row(s) affected	0.015 sec
2	04:16:55	use clinicas	0 row(s) affected	0.000 sec

Creamos la tabla médica

Solís Valle, Belén María: Perfil

FORMACIÓN PROFESIONAL

Área personal Mis cursos

Solís Valle, Belén María

Editar perfil

Información Personal

Dirección de correo:
moondream19@hotmail.com

País:
España

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

clinicas

Tables

medico

Columns

n_colegiado
nombre
especialidad

Indexes
Foreign Keys
Triggers
Views
Stored Procedures
Functions

sakila
sys
world

Administration Schemas

Information

Schema: clinicas

Object Info Session

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	03:52:38	create database if not exists clinicas	1 row(s) affected	0.015 sec
2	04:16:55	use clinicas	0 row(s) affected	0.000 sec
3	04:21:23	create table if not exists MEDICO (/*El campo n_colegiado debe almacenar valores entre 1 y 999999999.*/ n_colegiado varchar(9) primary key check (n_colegiado between 1 and 999999999), nombre varchar(50), especialidad varchar(25));	0 row(s) affected	0.047 sec

/*Creamos la tabla medico*/
create table if not exists MEDICO (
/*El campo n_colegiado debe almacenar valores entre 1 y 999999999.*/
n_colegiado varchar(9) primary key check (n_colegiado between 1 and 999999999),
nombre varchar(50),
especialidad varchar(25)
);

/*Creamos la tabla ciudad*/
create table if not exists CIUDAD (
/*El campo cp almacena un valor que puede estar entre 00240 y 90007.*/
cp char(5) primary key check (cp between 240 and 90007),
nombre varchar(25),
/*La ciudad más habitada del mundo es Tokio, Japón, con una población total de 37.340.000
el tipo de dato mediumint unsigned se queda en 16777215 por lo que no alcanzaría este tipo
es por ello que establecemos como dato el tipo int y unsigned */
num_habitantes int unsigned

Creamos la tabla ciudad

Solís Valle, Belén María: Perf

FORMACIÓN PROF

Área personal Mis cu

Solís Valle, Belén María

Editar perfil

Información Personal

Dirección de correo: moondream19@hotmail.com

País: España

MySQL Workbench

Local instance MySQL80 ×

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

clinicas

Tables

ciudad

Columns

cp

nombre

num_habitant

Indexes

Foreign Keys

Triggers

medico

Views

Stored Procedures

Functions

sakila

Administration Schemas

Information

Schema: clinicas

Object Info Session

Solis_Valle_Belen_Maria_BD_T_ ×

24 /*Creamos la tabla ciudad*/

25 • create table if not exists CIUDAD (

26 /*El campo cp almacena un valor que puede estar entre 00240 y 90007.*/

27 cp char(5) primary key check (cp between 240 and 90007),

28 nombre varchar(25),

29 /*La ciudad más habitada del mundo es Tokio, Japón, con una población total de 37.340.000

30 el tipo de dato mediumint unsigned se queda en 16777215 por lo que no alcanzaría este tipo

31 es por ello que estableceremos como dato el tipo int y unsigned */

32 num_habitantes int unsigned

33);

34

35 /*Creamos la tabla paciente*/

36 • create table if not exists PACIENTE(

37 nif char(9),

38 nombre varchar(50),

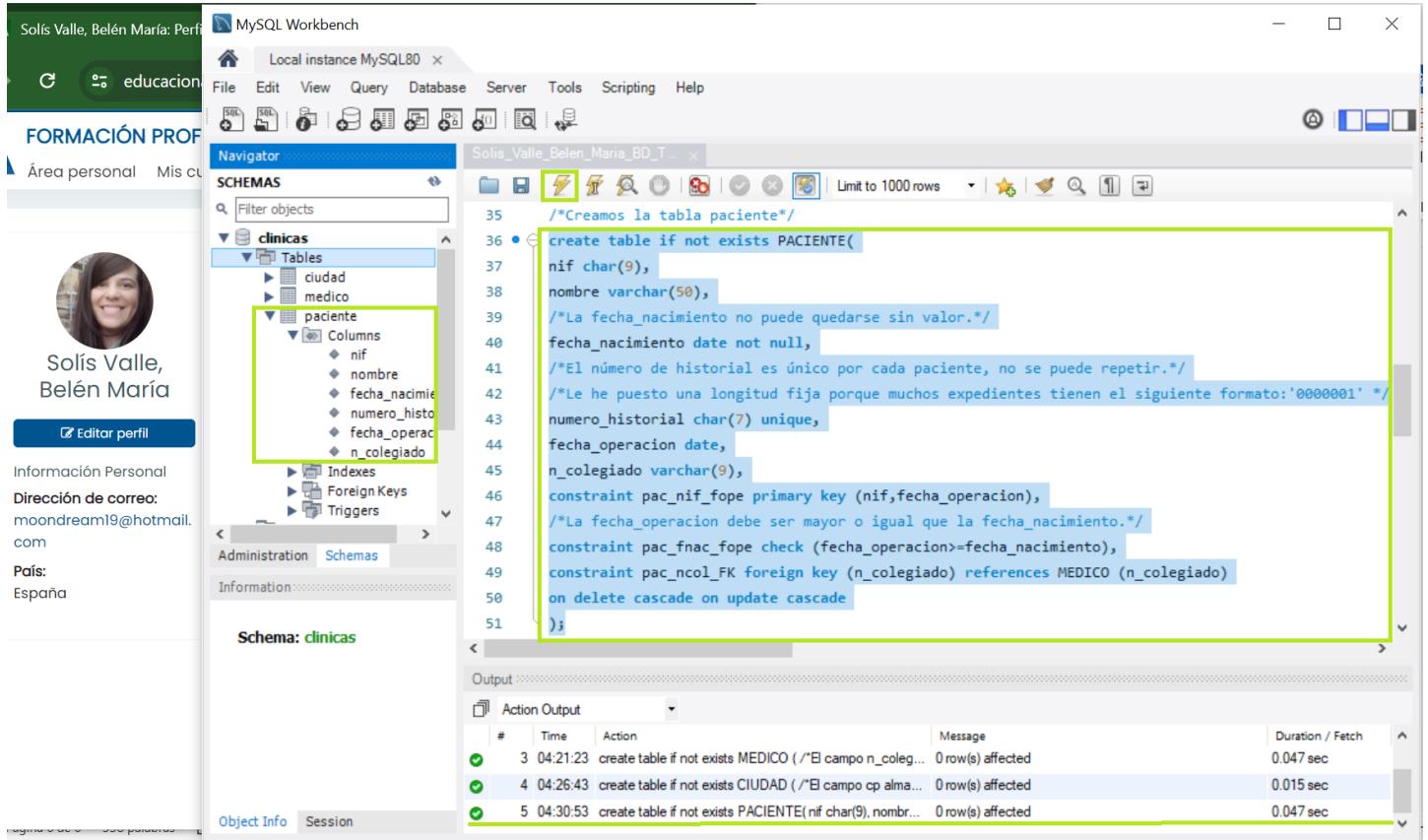
39 /*La fecha_nacimiento no puede quedar sin valor.*/

40 fecha_nacimiento date not null,

Action Output

#	Time	Action	Message	Duration / Fetch
2	04:16:55	use clinicas	0 row(s) affected	0.000 sec
3	04:21:23	create table if not exists MEDICO (/El campo n_coleg...	0 row(s) affected	0.047 sec
4	04:26:43	create table if not exists CIUDAD (/El campo cp alma...	0 row(s) affected	0.015 sec

Creamos la tabla paciente



Screenshot of MySQL Workbench showing the creation of the PACIENTE table. The code is highlighted in yellow.

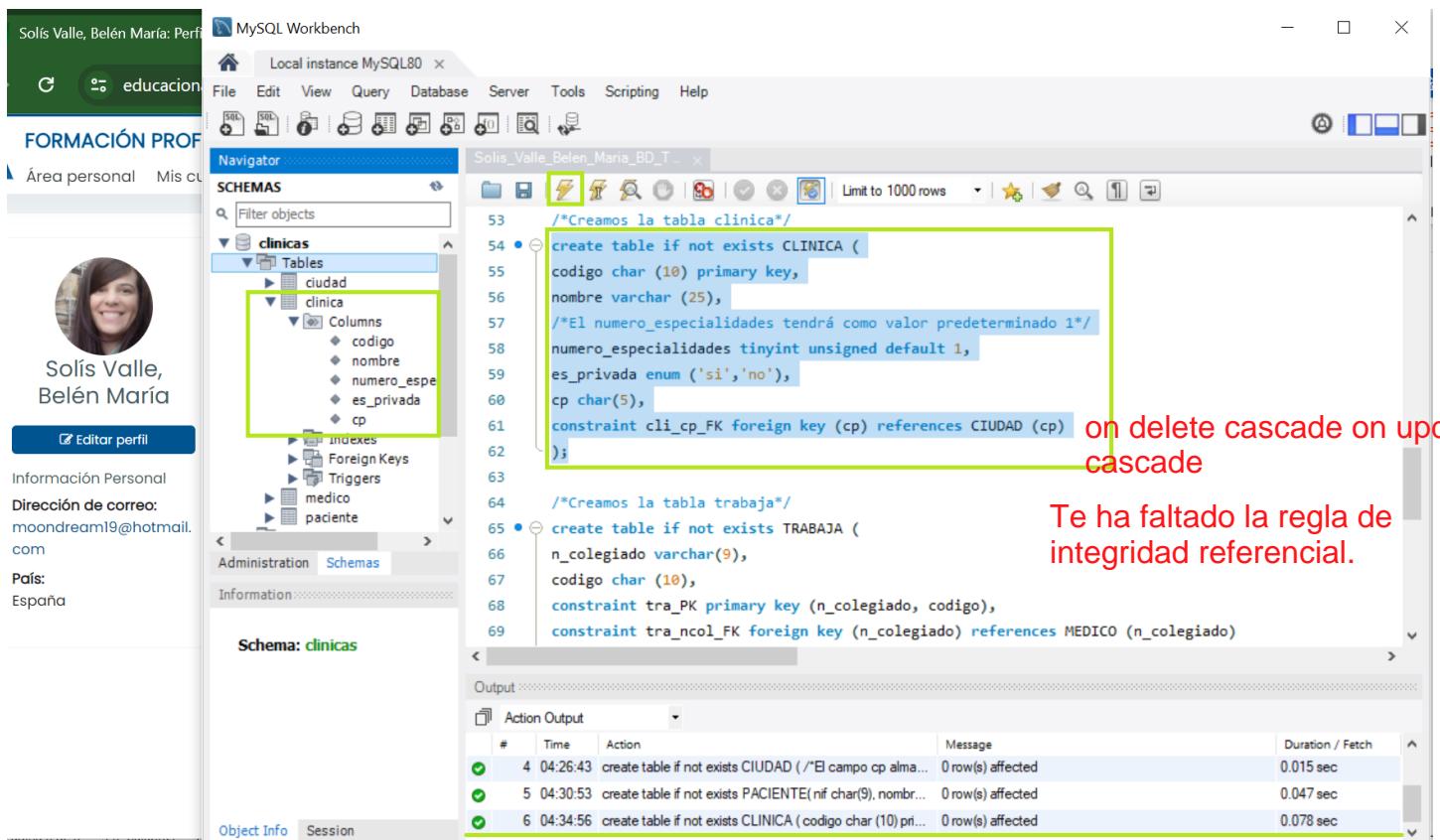
```

35  /*Creamos la tabla paciente*/
36  create table if not exists PACIENTE(
37      nif char(9),
38      nombre varchar(50),
39      /*La fecha_nacimiento no puede quedarse sin valor.*/
40      fecha_nacimiento date not null,
41      /*El número de historial es único por cada paciente, no se puede repetir.*/
42      numero_histolario char(7) unique,
43      /*Le he puesto una longitud fija porque muchos expedientes tienen el siguiente formato:'0000001' */
44      fecha_operacion date,
45      n_colegiado varchar(9),
46      constraint pac_nif_fope primary key (nif,fecha_operacion),
47      /*La fecha_operacion debe ser mayor o igual que la fecha_nacimiento.*/
48      constraint pac_fnac_fope check (fecha_operacion>=fecha_nacimiento),
49      constraint pac_ncol_fk foreign key (n_colegiado) references MEDICO (n_colegiado)
50          on delete cascade on update cascade
51  );

```

The screenshot shows the MySQL Workbench interface with the Navigator pane open, displaying the 'clínicas' schema and its tables. The 'paciente' table is selected. The SQL editor pane contains the code for creating the 'PACIENTE' table, which includes constraints for primary key, unique values, and referential integrity. The output pane shows the successful execution of the statements.

Creamos la tabla clínica



Screenshot of MySQL Workbench showing the creation of the CLINICA table. The code is highlighted in yellow. A red annotation points to the 'on delete cascade on update cascade' part of the constraint.

```

53  /*Creamos la tabla clinica*/
54  create table if not exists CLINICA (
55      codigo char (10) primary key,
56      nombre varchar (25),
57      /*El numero_especialidades tendrá como valor predeterminado 1*/
58      numero_especialidades tinyint unsigned default 1,
59      es_privada enum ('si','no'),
60      cp char(5),
61      constraint cli_cp_FK foreign key (cp) references CIUDAD (cp)
62  );

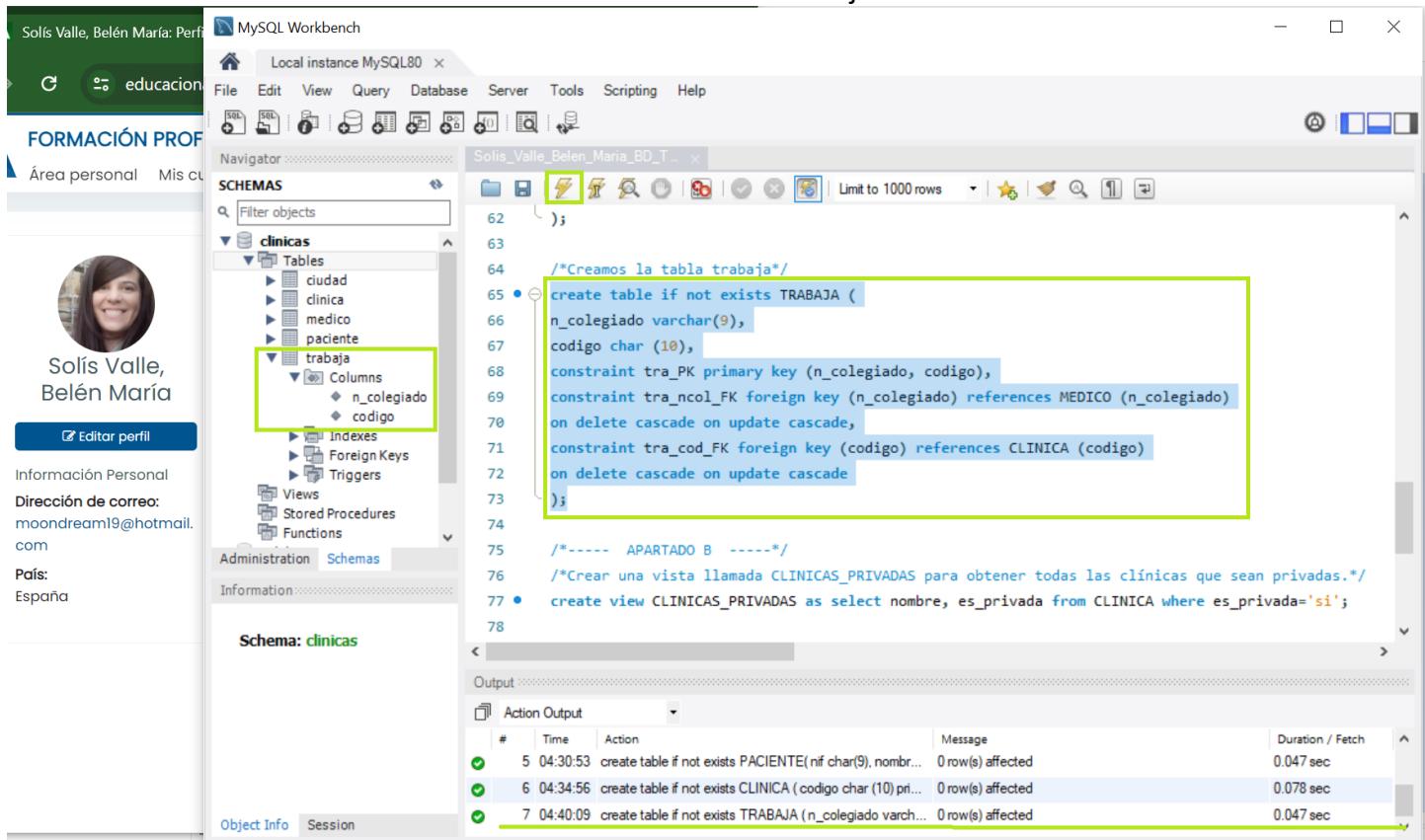
```

The red annotation highlights the 'on delete cascade on update cascade' part of the constraint definition. The screenshot shows the MySQL Workbench interface with the Navigator pane open, displaying the 'clínicas' schema and its tables. The 'clinica' table is selected. The SQL editor pane contains the code for creating the 'CLINICA' table, which includes a primary key and a foreign key constraint. The output pane shows the successful execution of the statements.

on delete cascade on update cascade

Te ha faltado la regla de integridad referencial.

Creamos la tabla trabaja



The screenshot shows the MySQL Workbench interface. On the left, the Navigator pane displays the 'clínicas' schema with its tables: ciudad, clínica, medico, paciente, and trabaja. The 'trabaja' table is selected and highlighted with a green box. Its columns, n_colegiado and codigo, are also highlighted. The SQL editor pane contains the DDL code for creating the 'trabaja' table, which includes primary key constraints and foreign key constraints linking to the 'MEDICO' and 'CLINICA' tables. The Output pane shows the execution log with three successful operations: creating the 'PACIENTE', 'CLINICA', and 'TRABAJA' tables.

```

/*
Creamos la tabla trabaja*/
create table if not exists TRABAJA (
n_colegiado varchar(9),
codigo char (10),
constraint tra_pk primary key (n_colegiado, codigo),
constraint tra_ncol_fk foreign key (n_colegiado) references MEDICO (n_colegiado)
on delete cascade on update cascade,
constraint tra_cod_fk foreign key (codigo) references CLINICA (codigo)
on delete cascade on update cascade
);

/*----- APARTADO B -----*/
/*Crear una vista llamada CLINICAS_PRIVADAS para obtener todas las clínicas que sean privadas.*/
create view CLINICAS_PRIVADAS as select nombre, es_privada from CLINICA where es_privada='si';

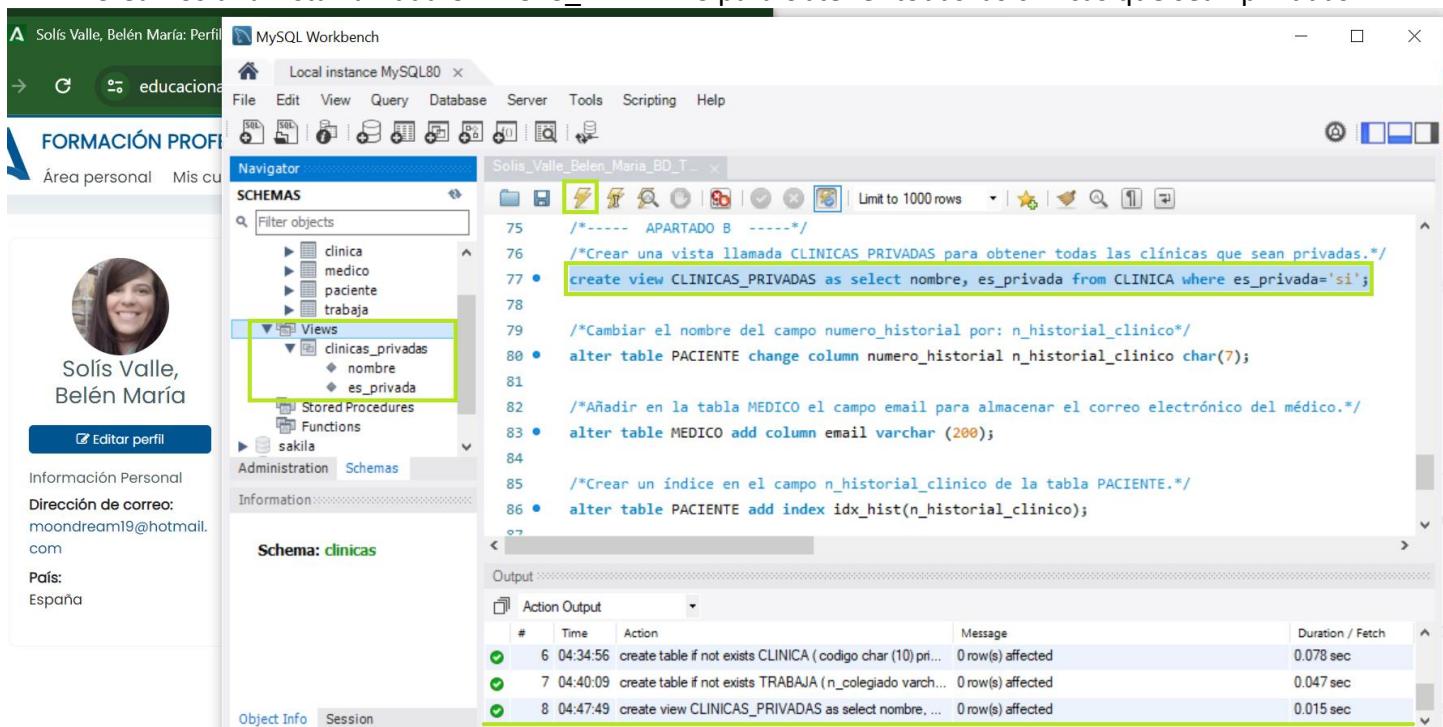
```

#	Time	Action	Message	Duration / Fetch
5	04:30:53	create table if not exists PACIENTE(nif char(9), nombr...	0 row(s) affected	0.047 sec
6	04:34:56	create table if not exists CLINICA (codigo char (10) pri...	0 row(s) affected	0.078 sec
7	04:40:09	create table if not exists TRABAJA (n_colegiado varch...	0 row(s) affected	0.047 sec

B) Despues de crear las tablas, vamos a realizar algunas modificaciones sobre ellas utilizando tambien el DDL.
Debes crear una sentencia SQL para cada subapartado que realice lo siguiente:

- 1) Crear una vista llamada CLINICAS_PRIVADAS para obtener todas las clínicas que sean privadas.
- 2) Cambiar el nombre del campo numero_historial por: n_historial_clinico
- 3) Añadir en la tabla MEDICO el campo email para almacenar el correo electrónico del médico.
- 4) Crear un índice en el campo n_historial_clinico de la tabla PACIENTE.

Creamos una vista llamada CLINICAS_PRIVADAS para obtener todas las clínicas que sean privadas.



The screenshot shows the MySQL Workbench interface. The Navigator pane displays the 'clínicas' schema with its tables: clínica, medico, paciente, and trabaja. A view named 'clinicas_privadas' is selected and highlighted with a green box. It has two columns: nombre and es_privada. The SQL editor pane contains the DDL code for creating the 'CLINICAS_PRIVADAS' view, changing the column name in the 'PACIENTE' table, adding the 'email' column to the 'MEDICO' table, and creating an index on the 'n_historial_clinico' column of the 'PACIENTE' table. The Output pane shows the execution log with four successful operations: creating the 'CLINICA', 'TRABAJA', 'PACIENTE' tables, and the 'CLINICAS_PRIVADAS' view.

```

/*
----- APARTADO B -----
/*Crear una vista llamada CLINICAS PRIVADAS para obtener todas las clínicas que sean privadas.*/
create view CLINICAS_PRIVADAS as select nombre, es_privada from CLINICA where es_privada='si';

/*Cambiar el nombre del campo numero_historial por: n_historial_clinico*/
alter table PACIENTE change column numero_historial n_historial_clinico char(7);

/*Añadir en la tabla MEDICO el campo email para almacenar el correo electrónico del médico.*/
alter table MEDICO add column email varchar (200);

/*Crear un indice en el campo n_historial_clinico de la tabla PACIENTE.*/
alter table PACIENTE add index idx_hist(n_historial_clinico);

```

#	Time	Action	Message	Duration / Fetch
6	04:34:56	create table if not exists CLINICA (codigo char (10) pri...	0 row(s) affected	0.078 sec
7	04:40:09	create table if not exists TRABAJA (n_colegiado varch...	0 row(s) affected	0.047 sec
8	04:47:49	create view CLINICAS_PRIVADAS as select nombre, ...	0 row(s) affected	0.015 sec

Cambiamos el nombre del campo numero_historial por: n_historial_clinico

```

72     on delete cascade on update cascade
73 );
74
75 /*----- APARTADO B -----*/
76 /*Crear una vista llamada CLINICAS_PRIVADAS para obtener todas las clínicas que sean privadas.*/
77 • create view CLINICAS_PRIVADAS as select nombre, es_privada from CLINICA where es_privada='si';
78
79 /*Cambiar el nombre del campo numero_historial por: n_historial_clinico*/
80 • alter table PACIENTE change column numero_historial n_historial_clinico char(7);
81
82 /*Añadir en la tabla MEDICO el campo email para almacenar el correo electrónico del médico.*/
83 • alter table MEDICO add column email varchar (200);
84
85 /*Crear un índice en el campo n_historial_clinico de la tabla PACIENTE.*/
86 • alter table PACIENTE add index idx_hist(n_historial_clinico);
87
88 /*----- APARTADO C -----*/

```

#	Time	Action	Message	Duration / Fetch
7	04:40:09	create table if not exists TRABAJA (n_colegiado varchar(200) ... 0 row(s) affected		0.047 sec
8	04:47:49	create view CLINICAS_PRIVADAS as select nombre, ... 0 row(s) affected		0.015 sec
9	04:55:18	alter table PACIENTE change column numero_historial ... 0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0 0.078 sec		

Añadimos en la tabla MEDICO el campo email para almacenar el correo electrónico del médico.

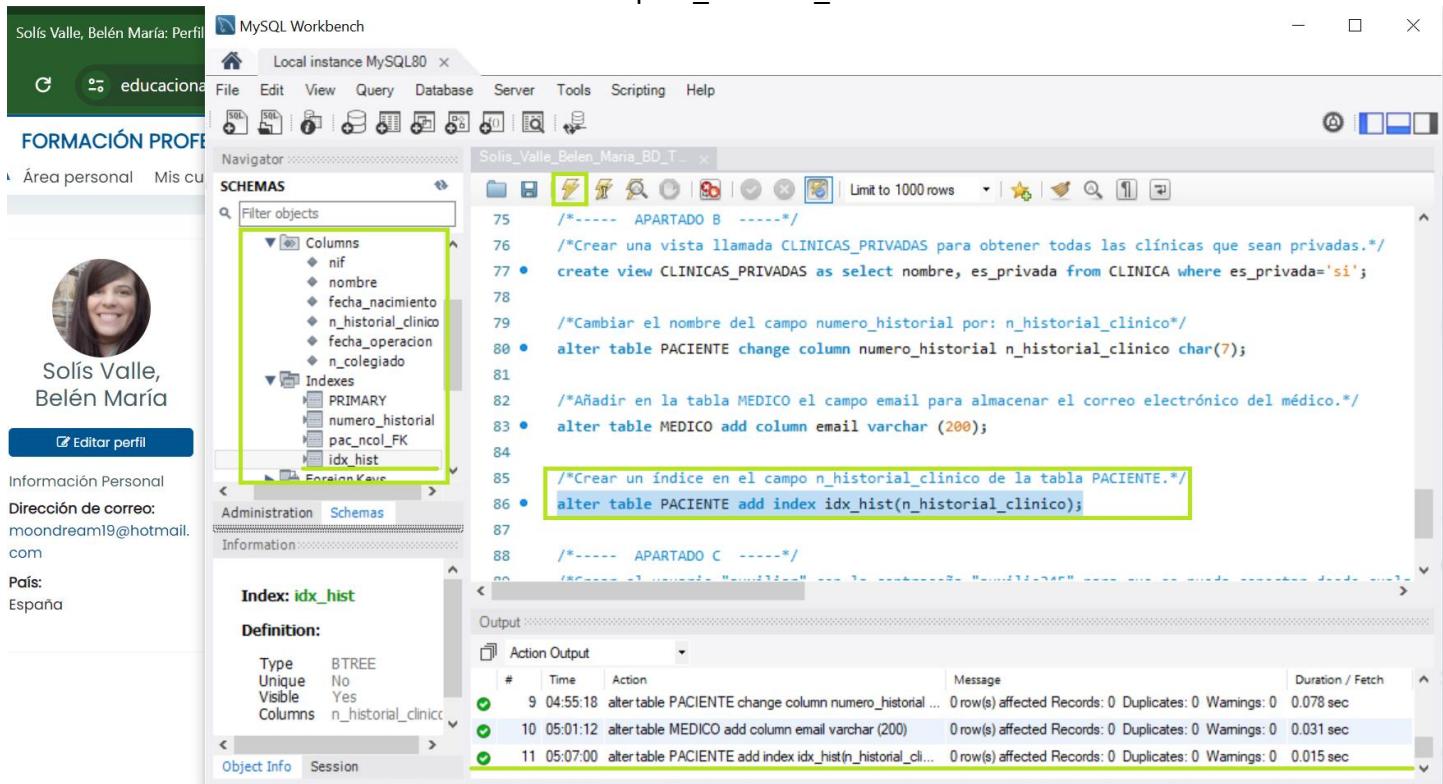
```

74
75 /*----- APARTADO B -----*/
76 /*Crear una vista llamada CLINICAS_PRIVADAS para obtener todas las clínicas que sean privadas.*/
77 • create view CLINICAS_PRIVADAS as select nombre, es_privada from CLINICA where es_privada='si';
78
79 /*Cambiar el nombre del campo numero_historial por: n_historial_clinico*/
80 • alter table PACIENTE change column numero_historial n_historial_clinico char(7);
81
82 /*Añadir en la tabla MEDICO el campo email para almacenar el correo electrónico del médico.*/
83 • alter table MEDICO add column email varchar (200);
84
85 /*Crear un índice en el campo n_historial_clinico de la tabla PACIENTE.*/
86 • alter table PACIENTE add index idx_hist(n_historial_clinico);
87
88 /*----- APARTADO C -----*/
89 /*Crear el usuario "auxiliar" con la contraseña "auxilio345" para que se pueda conectar desde cualquier dispositivo*/
90 • create user auxiliar@'%' identified by 'auxilio345';

```

#	Time	Action	Message	Duration / Fetch
8	04:47:49	create view CLINICAS_PRIVADAS as select nombre, ... 0 row(s) affected		0.015 sec
9	04:55:18	alter table PACIENTE change column numero_historial ... 0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0 0.078 sec		
10	05:01:12	altertable MEDICO add column email varchar (200) 0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0 0.031 sec		

Creamos un índice en el campo n_historial_clinico de la tabla PACIENTE



The screenshot shows the MySQL Workbench interface. On the left, there's a sidebar with user information (Solís Valle, Belén María) and profile settings. The main area shows the Navigator pane with the Schemas section open, displaying tables like clínicas, paciente, and MEDICO. The SQL editor tab contains a script with several SQL statements. The statement at line 86 is highlighted:

```

85 /*Crear un índice en el campo n_historial_clinico de la tabla PACIENTE.*/
86 alter table PACIENTE add index idx_hist(n_historial_clinico);
87
88 /*----- APARTADO C -----*/
89
90
91
92
93
94

```

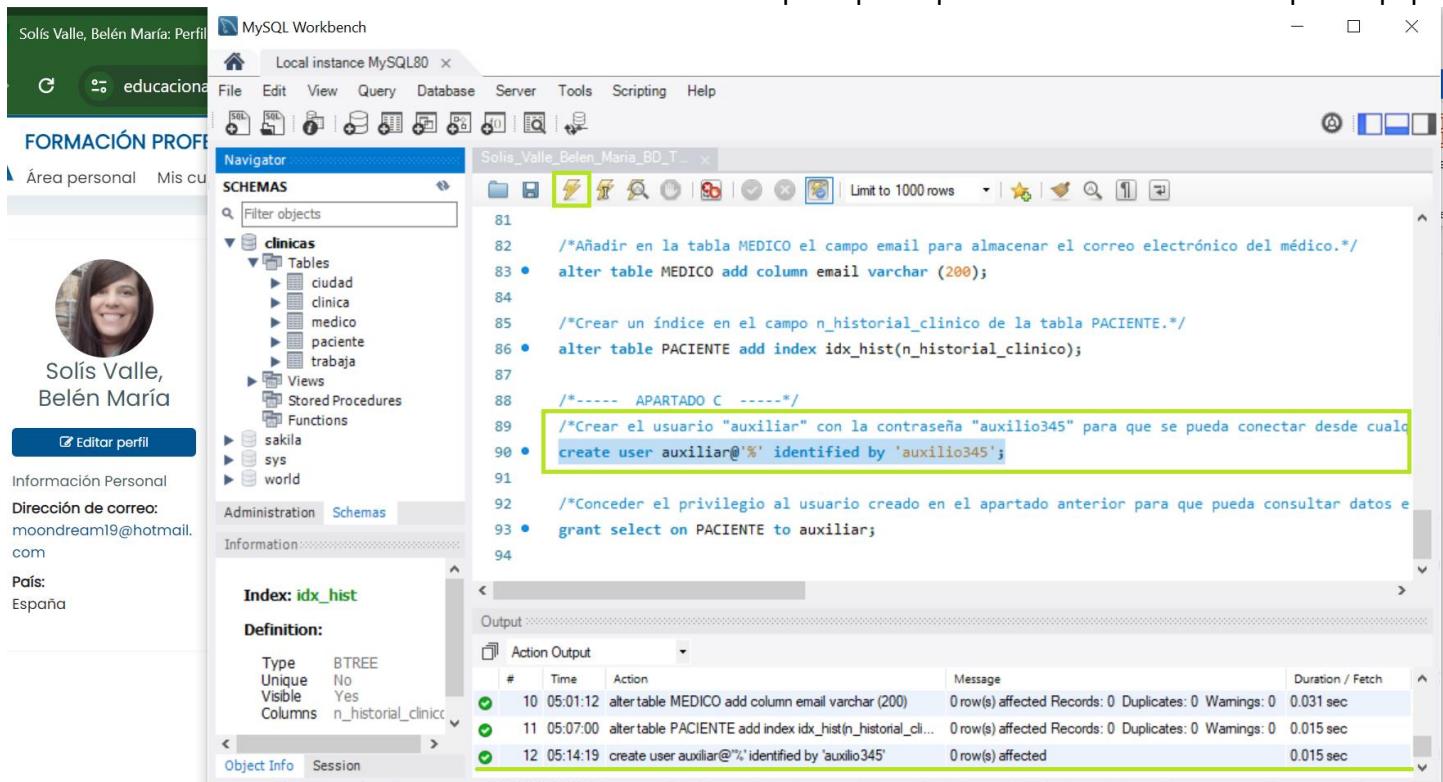
The 'Object Info' pane on the right shows details for the 'idx_hist' index, including its type (BTREE), unique status (No), visibility (Yes), and the single column it indexes (n_historial_clinico). The 'Action Output' pane at the bottom lists three actions with their timestamps and results.

#	Time	Action	Message	Duration / Fetch
9	04:55:18	alterable PACIENTE change column numero_historial ...	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.078 sec
10	05:01:12	alterable MEDICO add column email varchar (200)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.031 sec
11	05:07:00	alterable PACIENTE add index idx_hist(n_historial_cli...	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.015 sec

C) Por último, para gestionar la creación y permisos de usuarios, vamos a utilizar el lenguaje DCL. Debes crear una sentencia SQL para cada subapartado que realice lo siguiente:

- 1) Crear el usuario "auxiliar" con la contraseña "auxilio345" para que se pueda conectar desde cualquier equipo.
- 2) Conceder el privilegio al usuario creado en el apartado anterior para que pueda consultar datos en la tabla PACIENTE.

Creamos el usuario "auxiliar" con la contraseña "auxilio345" para que se pueda conectar desde cualquier equipo



The screenshot shows the MySQL Workbench interface. The Navigator pane shows the 'clinicas' schema with tables like ciudad, clinica, medico, paciente, and trabaja. The SQL editor tab contains a script with several SQL statements. The statement at line 90 is highlighted:

```

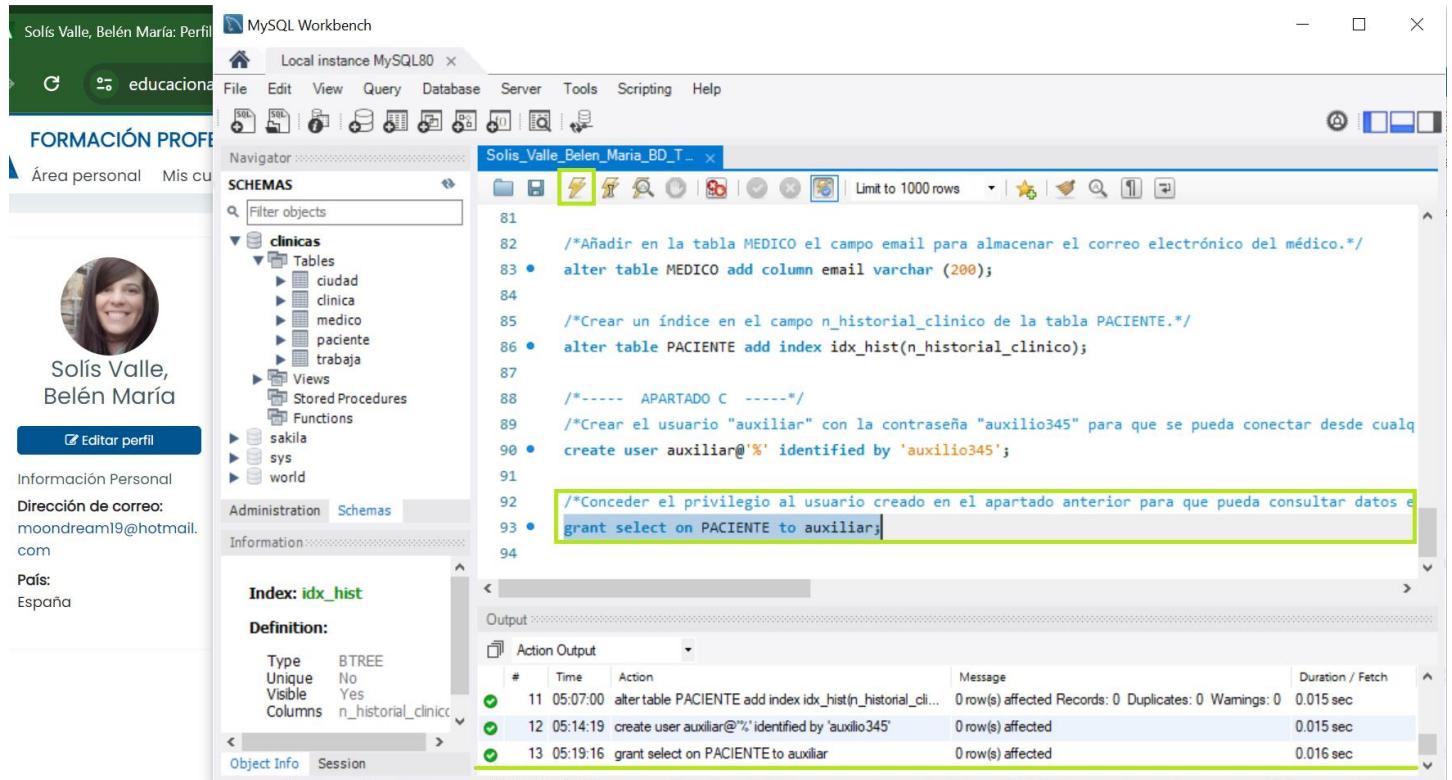
81
82 /*Añadir en la tabla MEDICO el campo email para almacenar el correo electrónico del médico.*/
83 alter table MEDICO add column email varchar (200);
84
85 /*Crear un índice en el campo n_historial_clinico de la tabla PACIENTE.*/
86 alter table PACIENTE add index idx_hist(n_historial_clinico);
87
88 /*----- APARTADO C -----*/
89 /*Crear el usuario "auxiliar" con la contraseña "auxilio345" para que se pueda conectar desde cualquier equipo*/
90 create user auxiliar@'%' identified by 'auxilio345';
91
92 /*Conceder el privilegio al usuario creado en el apartado anterior para que pueda consultar datos en la tabla PACIENTE*/
93 grant select on PACIENTE to auxiliar;
94

```

The 'Object Info' pane on the right shows details for the 'idx_hist' index. The 'Action Output' pane at the bottom lists four actions with their timestamps and results.

#	Time	Action	Message	Duration / Fetch
10	05:01:12	alterable MEDICO add column email varchar (200)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.031 sec
11	05:07:00	alterable PACIENTE add index idx_hist(n_historial_cli...	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.015 sec
12	05:14:19	create user auxiliar@'%' identified by 'auxilio345'	0 row(s) affected	0.015 sec

Concedemos el privilegio al usuario creado para que pueda consultar datos en la tabla PACIENTE



The screenshot shows the MySQL Workbench interface. On the left, there's a sidebar with user information (Solís Valle, Belén María: Perfil, Área personal, Mis cursos) and a profile picture. The main area has a 'Navigator' pane showing 'SCHEMAS' (clinicas, sakila, sys, world) and a 'Schemas' tab selected. A script editor window titled 'Solis_Valle_Belen_Maria_BD_T...' contains the following SQL code:

```

81
82  /*Añadir en la tabla MEDICO el campo email para almacenar el correo electrónico del médico.*/
83 • alter table MEDICO add column email varchar (200);
84
85 /*Crear un índice en el campo n_historial_clinico de la tabla PACIENTE.*/
86 • alter table PACIENTE add index idx_hist(n_historial_clinico);
87
88 /*----- APARTADO C -----*/
89 /*Crear el usuario "auxiliar" con la contraseña "auxilio345" para que se pueda conectar desde cualquier parte*/
90 • create user auxiliar@'%' identified by 'auxilio345';
91
92 /*Conceder el privilegio al usuario creado en el apartado anterior para que pueda consultar datos en la tabla PACIENTE*/
93 • grant select on PACIENTE to auxiliar;
94

```

The line 'grant select on PACIENTE to auxiliar;' is highlighted with a yellow box. Below the script editor is a 'Log' viewer showing the execution history:

#	Time	Action	Message	Duration / Fetch
11	05:07:00	alter table PACIENTE add index idx_hist(n_historial_cli... 0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.015 sec	
12	05:14:19	create user auxiliar@'%' identified by 'auxilio345' 0 row(s) affected	0.015 sec	
13	05:19:16	grant select on PACIENTE to auxiliar 0 row(s) affected	0.016 sec	