



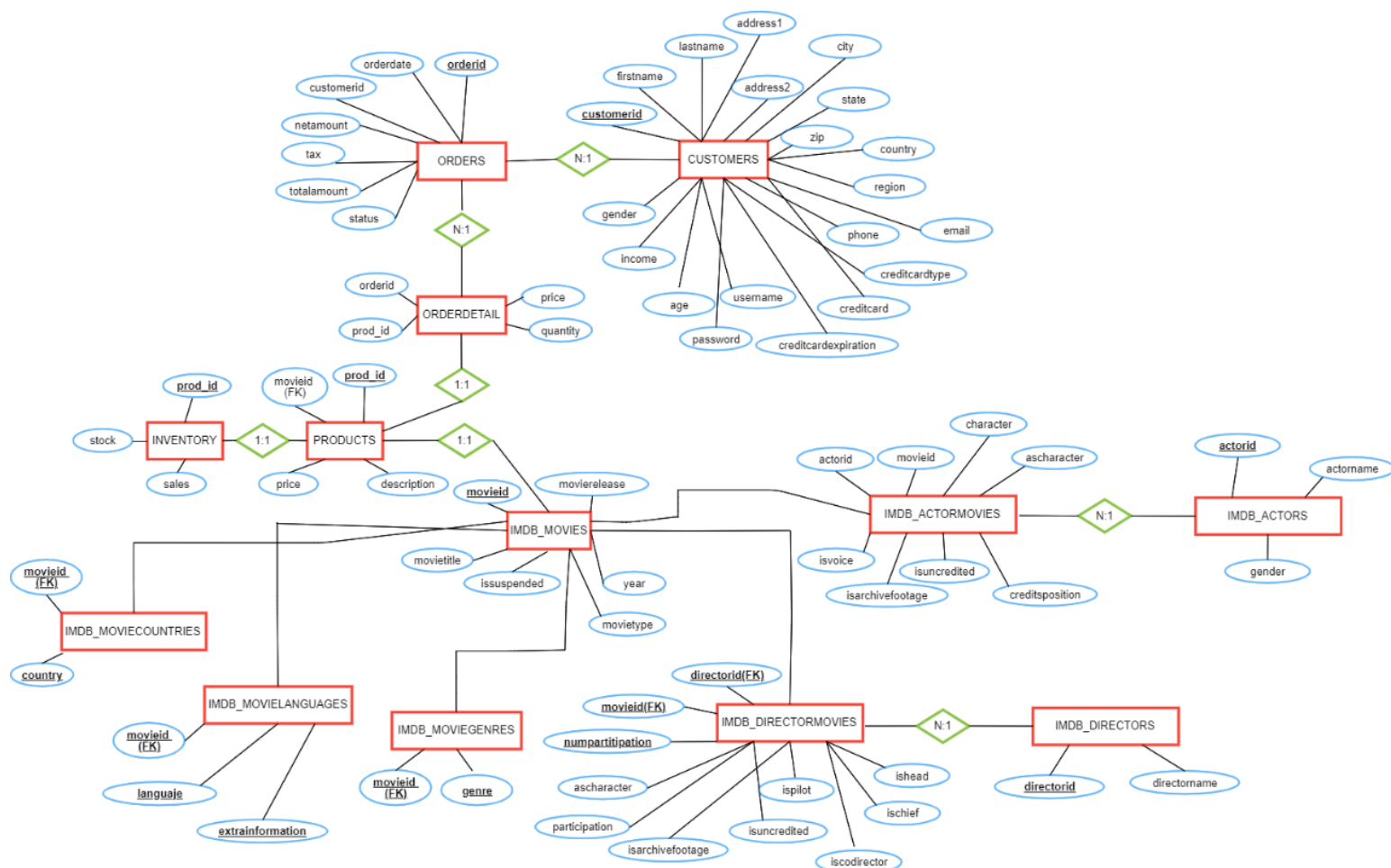
Memoria P2

Sistemas Informáticos. Curso
2020/2021

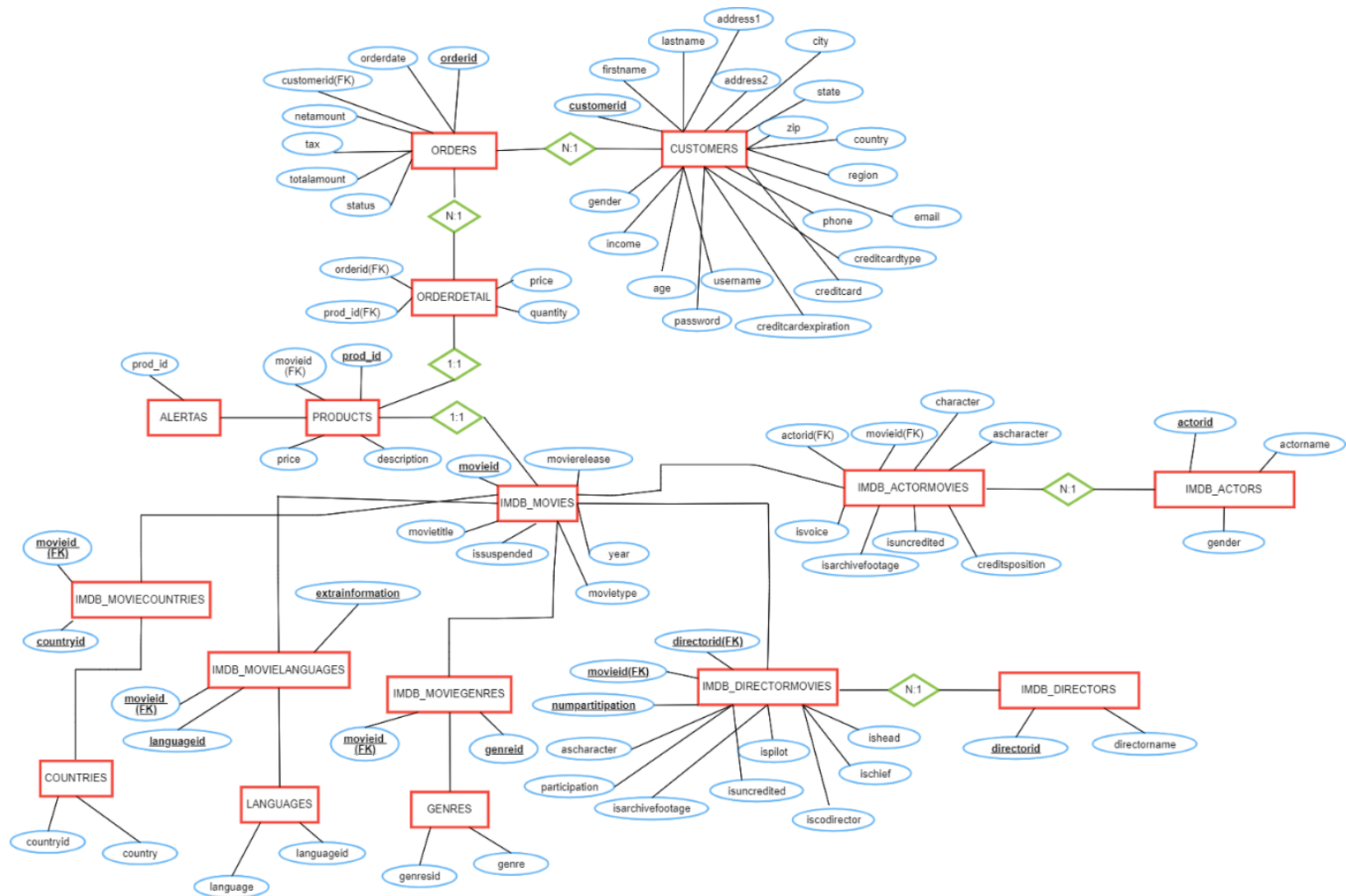
Adrian San Felipe Martin y Luis Miguel Duran Diaz

DIAGRAMAS ENTIDAD RELACIÓN ANTES Y DESPUÉS

Al desplegar la base de datos sin realizar ninguna modificación y realizar ingeniería inversa, el diagrama entidad relación que obtenemos es el siguiente (Las columnas que están subrayadas y en negrita son clave primaria mientras que clave foránea son las que tienen "(FK)":



La base de datos después de aplicar actualiza.sql es la siguiente:



Como se puede observar, hemos añadido las tablas de alertas, countries, languages y genres, con sus respectivas columnas. También hemos añadido claves foráneas que después de discutirlo veíamos que tenía sentido incluir en determinadas tablas. Hay otros cambios que también se reflejan en el diagrama entidad relación, como la actualización de los nombres de las columnas de las tablas imdb_moviecountries, imdb_movelanguages y imdb_moviegenres.

CONSULTAS, TRIGGERS, FUNCIONES SOLICITADAS Y ARCHIVOS

- **actualiza.sql:** Este archivo sql debe ser ejecutado justo después de desplegar la base de datos proporcionada, ya que actualizará la base de datos con nuevas características que detallaremos a continuación. Creamos la tabla alertas, que almacenará en ella el id del producto cuando no quede stock del mismo. También hemos alterado diferentes tablas poniendo que algunas de sus columnas sean NOT NULL (por ejemplo, la columna year de la tabla imdb_movies, la columna customerid de orders, etc) y hemos alterado varias columnas de diferentes tablas añadiendo claves foráneas (por ejemplo, la columna customerid de la tabla orders). Hemos actualizado también el valor del precio de la tabla orderdetail, poniéndolo a cero. Por otro lado, hemos decidido eliminar la tabla inventory, y añadir los campos de stock y sales a la tabla products (como en la tabla inventory antes había id de productos que no estaban “registrados” porque no tenían ni stock ni sales, en esos ids de producto ahora saldrá un 0 como valor por defecto, si no saldría NULL). Observamos también que en la base de datos existen nombres de usuario que son iguales, aunque sean distintas personas. Para solucionarlo hemos concatenado el nombre de usuario y su email, y para que no vuelva a suceder este problema, ponemos username como parámetro UNIQUE. Para finalizar, hemos creado nuevas entidades relacionadas con la tabla movies (countries, genres y languages) para garantizar la integridad de los datos y convertir los atributos multivaluados.
- **setPrice.sql:** Esta consulta actualiza los precios de los productos añadiéndoles el 2% de su precio cada año, por lo que, si una película salió a la venta en 2010, desde ese año en adelante se incrementará su precio un 2%. Usamos un CAST para formatear el número de decimales del precio de los productos usando FM9999.9999. Con esto lo que permitimos es esa misma cantidad de números y con FM eliminamos los ceros o espacios en blanco que se puedan generar. Con CAST también transformamos el tipo de price a float y to_char convierte la fecha de ordersdate de tipo date a char.
- **setOrderAmount.sql:** En este sql creamos una función setOrderAmount que se encarga de rellenar totalamount y netamount de la tabla de orders. Consta de dos consultas. La primera de ellas rellena netamount sumando el precio por la cantidad de veces que se ha comprado un producto, teniendo en cuenta que un producto puede ser comprado varias veces. Para actualizar totalamount obtenemos el precio de netamount y lo multiplicamos por los impuestos. Más tarde, hacemos un select de la función para comprobar que el código proporcionado se ejecuta correctamente.
- **getTopVentas.sql:** En este archivo sql crearemos una función getTopVentas, que recibe dos años distintos, primero el menor y luego el mayor de los años, y devuelve en una tabla las películas más vendidas entre los años introducidos, es decir, si se ejecuta getTopVentas(2014, 2020) se mostrará por pantalla la película más vendida en 2014, 2015... hasta 2020. La función tiene una subconsulta que selecciona el year y le pone como apodo ‘año’, movietitle de imdb_movies al que ponemos como apodo ‘título’ y la suma de las cantidades como ‘ventas’. Hacemos uso de ROW_NUMBER para poder partir la query en varios grupos, haciendolo por los años y ordenando por las ventas y el título de la película. Gracias a inner join, juntamos varias tablas de la base de datos por el id de la película, del producto y del pedido siempre que se de la condición de que el año de las películas se encuentre entre los dos parámetros que hemos introducido anteriormente. Agrupamos por título de la película y por fecha y los ordenamos por el año de manera ascendente y las ventas de manera descendente. Gracias a la función de ROW_NUMBER de la subconsulta anterior, nos habrá quedado una tabla ordenada por este parámetro, donde la película más vendida de ese año tendrá como ROW_NUMBER=1, por lo que ponemos esa condición y ordenamos por ventas y por nombre de película.

```

si1=# \i sql/getTopVentas.sql
DROP FUNCTION
CREATE FUNCTION
anio |          pelicula          | ventas
-----+-----+-----
2017 | Illtown (1996)             | 142
2016 | Love and a .45 (1994)      | 136
2019 | Life Less Ordinary, A (1997) | 134
2018 | Wizard of Oz, The (1939)    | 134
2015 | No Looking Back (1998)      | 101
2020 | Gang Related (1997)         | 57
2014 | Male and Female (1919)      | 9
(7 filas)

```

- getTopMonths.sql:** En este archivo sql declaramos una función getTopMonths que recibe dos valores big int 'num_prod' y 'max_amount', el primero es el umbral de numero de productos y el segundo es el importe. Necesitamos las tablas de orders y orderdetail para sacar el totalamount y la cantidad, y las relacionaremos por el orderid. Mas tarde, las agrupamos y ordenamos por año y mes gracias a la funcionalidad de date_part y tendrá que cumplir la condición de que el totalamount o la suma de las cantidades sean mayores que los datos introducidos al ejecutar la función. Al final, ordenamos por año y ejecutamos la función con los datos indicados en el guión de la práctica. Cuando ejecutamos la función nos devuelve 68 filas, en la siguiente imagen encontramos unos cuantos de esos resultados:

anio	mes	importe	productos
2015	1	462474.5459	3881
2015	2	550242.6872	4629
2015	3	864630.7187	7262
2015	4	935437.0202	7897
2015	5	1189232.4829	10146
2015	6	1313429.9973	11078
2015	7	1479052.1304	12547
2015	8	1759112.5324	14916
2015	9	1853862.5229	15639
2015	10	2131597.9447	18178
2015	11	2140980.1686	17917
2015	12	2071184.3150	17671
2016	1	2274247.9949	18867
2016	2	2041780.3065	16942
2016	3	2239910.7184	18302
2016	4	2128222.9948	17642
2016	5	2185313.7642	18194
2016	6	2099330.1793	17366
2016	7	2299889.6195	18760
2016	8	2263604.1589	18662
2016	9	2325144.8931	19133
2016	10	2292351.3602	19086
2016	11	2245447.9723	18329
2016	12	2300557.8859	18789
2017	1	2243565.4337	18252
2017	2	2165606.4087	17437
2017	3	2328214.7770	18713
2017	4	2121043.7743	17409
2017	5	2311697.4769	18915
2017	6	2291889.0254	18495
2017	7	2243859.1043	18214
2017	8	2276949.0045	18503
2017	9	2243290.0649	18148
2017	10	2300697.8820	18745
2017	11	2259004.4643	18273
2017	12	2332592.9228	18684
2018	1	2299209.3240	18394
2018	2	2178598.8190	17173
2018	3	2333588.6999	18683

- **updOrders.sql:** En este sql declaramos un trigger tal y como se nos ha enseñado en clase. Existen 3 diferentes casos y en base a estos la función es una u otra.
Si la opción del trigger es borrar, entonces cambiamos netamount quitando precio * cantidad. También se cambia totalamount, ya que al ser diferente netamount, esta cambiará en consecuencia.
Si la opción del trigger es insertar entonces en netamount se suma el precio por la cantidad de los nuevos productos insertados, y de nuevo se cambia totalamount con el valor que hayamos obtenido anteriormente de netamount.
Si la opción del trigger es actualizar, primero cambiamos el precio de netamount a los productos viejos, y luego se le suman los nuevos precios. Como en las otras dos opciones, totalamount cambiará en consecuencia con netamount.
- **updInventory.sql:** Si TG_OP es 'update', si el estado nuevo no es nulo y el estado antiguo es nulo, entonces en cada cantidad del id del producto que se compra se suma uno a las ventas y se resta uno al stock en la tabla products (uno o las cantidades que se han comprado). Para saber que productos se han quedado sin stock declaramos una variable stock_, que cuando sea igual a 0 introduzca en una nueva tabla 'alertas' el id del producto que se ha agotado.
- **database.py:** En este archivo de Python se encuentran las funciones necesarias para realizar la conexión con la base de datos gracias a sqlalchemy, para, por ejemplo, logearse con un usuario de la base de datos, almacenar la información de registro de un usuario en la base de datos, u obtener las películas mas vendidas en un año.

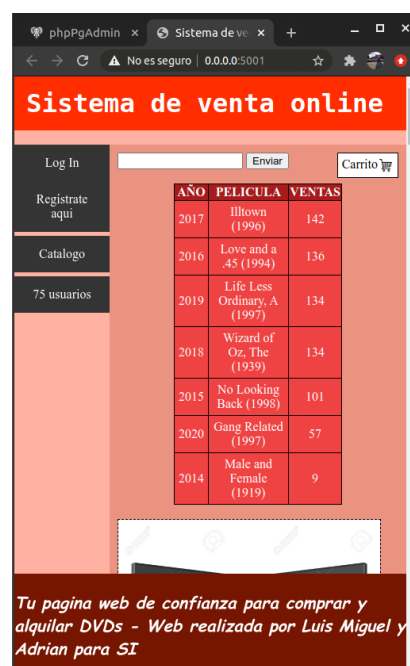
Existen más archivos, aunque los más importantes son los enunciados anteriormente, los otros archivos se encontraban también en la practica anterior y sobre ellos se han realizado modificaciones mínimas o ni siquiera modificaciones, como los HTML para la página web, imágenes necesarias para la página o archivos CSS.

PÁGINA WEB Y EJEMPLOS DE SU FUNCIONAMIENTO

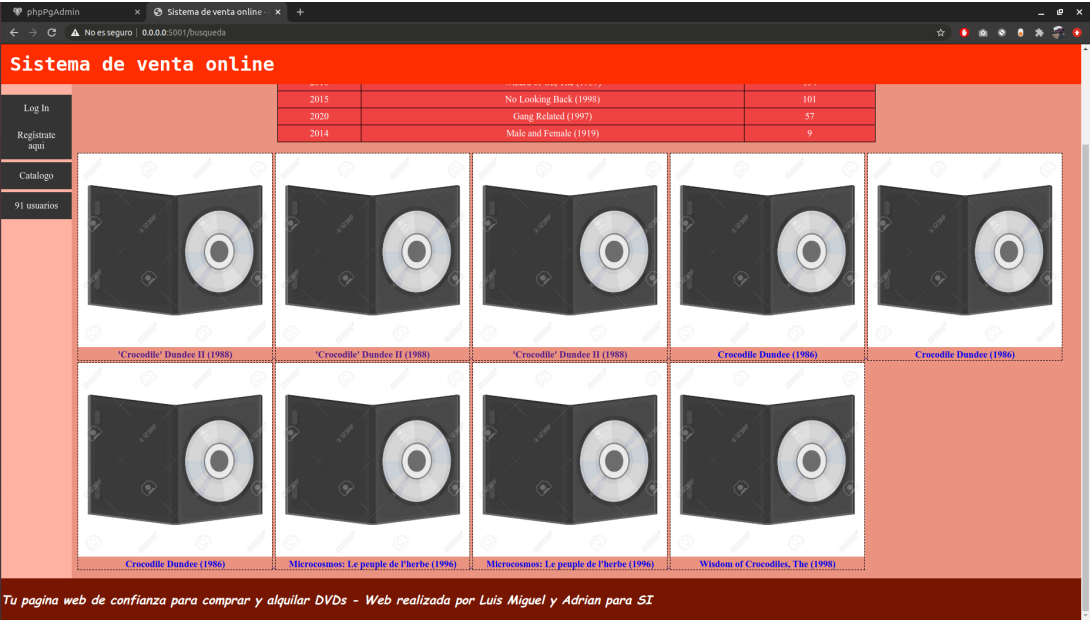
Al entrar en la página web después de haber ejecutado los anteriores sql nos encontramos con la siguiente información:



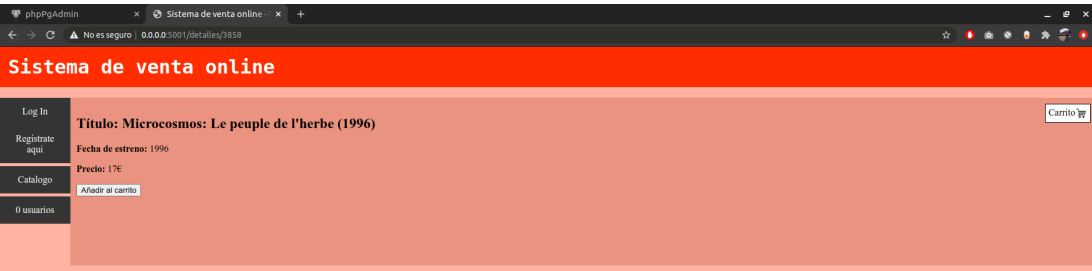
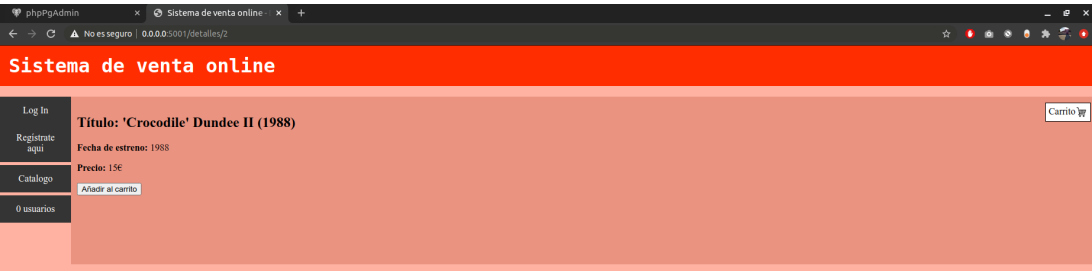
Como se puede observar en la imagen anterior, en la página principal se mostrará una tabla con las películas más vendidas de cada año, en la que se puede observar el año, el título de la película y cuantas se vendieron. Esto se consigue gracias a la función `getTopVentas` y a la conexión con `sqlalchemy` a la base de datos. En la página principal también podemos observar una lista de películas que también se obtienen de la base de datos. En este caso, hemos elegido que se muestren 20 películas. Como información extra, esta y todas las páginas de nuestro portal web tienen redimensionamiento dinámico, es decir, se redimensionan según ocupe la ventana, como se muestra en estos ejemplos:



Si pinchamos en una de las películas que se muestran en la página principal, nos lleva a ver más detalles de esa película. En la página principal también es posible introducir una cadena de texto para buscar una película específica. El buscador transforma la cadena introducida para que busque todas las películas que tengan esa cadena, ya sea en minúsculas o en mayúsculas. En el siguiente ejemplo se ha introducido en el buscador “Croco”.



Y como hemos comentado anteriormente, se puede pinchar en la imagen de la película para que te lleve a sus detalles.



A continuación, vamos a realizar pruebas de inicio de sesión y registro en la página web, empezando por lo primero. Para ello, nos dirigimos a la base de datos y escogemos un usuario al azar, por ejemplo, el usuario “benton” con contraseña “boil”, aunque se podría realizar con cualquier otro usuario.

The screenshot shows the phpPgAdmin interface. On the left, a tree view shows the database structure: **PostgreSQL** > **public** > **customers**. The main area displays the **customers** table with columns: **customerid**, **firstname**, **lastname**, **address1**, **address2**, **city**, **state**, **zip**, **country**, **region**, **email**, **phone**, **creditcardtype**, **creditcard**, **creditcardexpiration**, **username**, **password**, **age**, **income**, **gender**. The table contains 30 rows of user data. The first row is for user 'benton' with password 'boil'.

customerid	firstname	lastname	address1	address2	city	state	zip	country	region	email	phone	creditcardtype	creditcard	creditcardexpiration	username	password	age	income	gender
31	primed	but	faule fury 77	cranner hefter	prossa	bolary	58878	Portugal	delta	primed.but@gmail.com	+4 91272271	VISA	43965415252613	202411	benton	boil	45	2323 F	
32	clomp	andie	canopy bene 234	aria roden	worm	colony	40274	Senegal	bevel	clomp.andie@gmail.com	+1 18078834	Discover	64351432628789	202353	domani	boise	36	30207 M	
33	rotted	mood	hong mutant 253	arroyo rima	blimg	fushou	19027	Algeria	arline	rotted.mood@gmail.com	+38 28470432	Diners	422696465389192	202312	damp	arvil	51	10227 M	
34	kef	half	kefo hosa 213	lowell usary	cring	ring	36046	Zaire	codale	kef.half@gmail.com	+44 96020571	VISA	456246451198327	202306	isavan	jeak	33	31636 M	
35	lola	rbic	file alive 45	sorted elitch	basile	ape	96386	Peru	delune	lola.rbic@gmail.com	+38 53310593	VISA	456775371516839	202508	dimple	surges	33	17860 F	
36	payday	bali	elmer rum 246	reflex nikel	upton	reloff	23838	China	rafthe	payday.bali@gmail.com	+28 978762176	VISA	430108287918584	202210	plama	janice	34	48793 M	
37	anul	bonne	carrie fcaud 213	argo alupa	vidia	lagasu	28771	Bulgaria	marcar	anul.bonne@gmail.com	+52 84734346	VISA	4790847210219031	202405	posia	plus	45	21190 F	
38	paaw	ent	stake kusa 139	arawp prime	primr	garmon	27771	China	scoreel	paaw.ent@gmail.com	+54 596347199	VISA	407737345368877	202307	hali	wend	52	36354 M	
39	diad	ar	most kaven 166	drop water	anyone	blash	32693	Un	visonr	diad.ar@gmail.com	+1 13050172	VISA	459746333660904	202302	ezager	breed	47	26100 F	
40	reflex	whelp	crozan awe 172	randie whany	mundt	enr	49486	Netherlands	edam	reflex.whelp@gmail.com	+53 540388793	Mastercard	443207800657458	202506	bic	iswert	53	41101 M	
41	chavez	ilee	nguyen street 176	angia due	comde	astid	24513	Iran	musum	chavez.ilee@gmail.com	+35 981196889	American	4622477428841854	202211	loume	cecl	24	37492 M	
42	alexy	hoi	giving mail 199	pat kuling	basiah	backtel	15832	Italy	breil	alexy.hoi@gmail.com	+29 23376336	VISA	48318247511734	202305	caned	seka	38	46193 M	
43	lyoff	neadle	figurn rich 107	cherty amor	djan	bassef	33507	Cameroon	nashtu	lyoff.neadle@gmail.com	+17 627607108	Discover	46422362846672	202410	tenure	chrood	47	22372 M	
44	briggs	daniel	lethom otter 112	copi leas	isuz	quench	19197	Cameroon	nada	briggs.daniel@gmail.com	+12 932985271	American	429347389026813	202307	kulla	gri	28	54339 M	
45	relex	elie	tyle urawo 290	riker carla	quale	nush	50149	Italy	armon	relex.elie@gmail.com	+25 533712725	Diners	429558171668464	202409	benito	terrald	49	24422 M	
46	deavis	breed	Franki walton 146	gilly roach	diolan	boule	17303	Philippines	pehyr	deavis.breed@gmail.com	+24 163260609	Mastercard	49946865611354	202404	seval	seave	41	12978 M	
47	bitrap	weyden	bebop saab 67	joth crop	neut	unlot	14033	Poland	nush	bitrap.weyden@gmail.com	+3 276231641	VISA	46001448505650	202404	univac	potls	46	22212 M	
48	reflex	assort	whosr mar 150	snook shade	pufted	planck	49607	South Africa	lewis	reflex.assort@gmail.com	+19 370320734	Diners	4150903436871976	202312	elle	trade	100	17914 M	
49	tramp	impale	day fural 98	ava magic	helena	copite	12248	Canada	aspre	tramp.impale@gmail.com	+18 971365578	Mastercard	4600097152427648	202309	malyn	rubber	22	18006 F	
50	mental	ajonal	emma malthe 68	jetum uale	blash	manra	48031	Ireland	eddy	mental.ajonal@gmail.com	+13 70762481	American	47536446627961	202359	mayr	indare	53	48674 M	
51	primer	untl	betty pagno 130	wid chonic	ing	lef	35162	Argentina	gauchr	primer.untl@gmail.com	+20 142321496	Mastercard	4616964059323710	202410	whence	tracy	28	17171 M	
52	mograw	gibson	paling doyen 109	elrys alamo	duffer	rapped	15201	Switzerland	hub	mograw.gibson@gmail.com	+5 78379486	VISA	4072162637270413	202201	esiquen	quae	32	55040 M	
53	space	spil	metrol glkor 15	polux anbia	mtz	weir	34652	Italy	heny	space.spil@gmail.com	+51 286767431	American	426851841794881	202312	maris	meky	35	24599 F	
54	lake	said	inner naria 53	gowat aktum	shide	clann	47253	Russia	abodch	lake.said@gmail.com	+14 420454548	American	459262113874889	202511	rebed	placy	35	15158 M	
55	bare	ricky	loop freud 287	rsa fugat	nice	nuffy	52771	Senegal	wharf	bare.ricky@gmail.com	+55 15004530	American	450771212548735	202507	tau	gne	44	14680 F	
56	hel	raw	zippy peocoe 221	albi high	seeld	armon	21959	Hong Kong	nabn	hel.raw@gmail.com	+16 275732379	American	4474243416918968	202404	geni	oni	36	44191 F	
57	chill	janos	anla leant 8	blaz crease	oslo	coaly	55775	Greece	cancel	chill.janos@gmail.com	+37 650628274	American	498790326984024	202407	nadia	condom	28	51293 M	
58	side	cora	skut decay 114	egg skull	taylor	ama	37211	France	kama	side.cora@gmail.com	+39 599163275	American	4492645615042741	202402	dearie	byrce	23	46742 F	
59	vida	donna	punny meany 155	lophyi made	alumni	kora	24577	Russia	prop	vida.donna@gmail.com	+2 528104280	Mastercard	49633362678947	202201	edward	indus	28	40873 M	
60	sap	null	bookmy kane 62	bing peepy	rumple	enep	18936	Bulgaria	shuaf	sap.null@gmail.com	+19 950671867	Diners	484523593739564	202205	tom	cayy	54	44899 F	

Al iniciar sesión, nos lleva al historial de compra del usuario:

The screenshot shows the 'Sistema de venta online' login page. The login form has fields for 'Nombre de usuario:' (benton) and 'Contraseña:' (boil). The 'Log In' button is highlighted. Below the login form, there is a link to 'Historial de benton'. The 'Historial de benton' page shows a table with columns: Fecha, Importe, and Detalles. The table is empty. A modal dialog box is open, asking '¿Quieres guardar la contraseña?' (Do you want to save the password?). The dialog has fields for 'Nombre de usuario:' (benton) and 'Contraseña:' (boil). The 'Guardar' button is highlighted.

Fecha	Importe	Detalles
-------	---------	----------

Ahora vamos a registrarnos en la página web mediante el formulario disponible:

Sistema de venta online

Log In

Regístrate aquí

Catalogo

60 usuarios

Regístrate aquí

Nombre de usuario:
adrian

Contraseña:

Repite la contraseña:

E-mail:
adrianfe1@gmail.com

Nombre:
Adrian

Apellidos:
San Felipe

Edad:
24

Dirección:
Capitan Francisco Sanch

Dirección 2:

Ciudad:
Alcobendas

Estado:
Madrid

Codigo Postal:
28100

País:
España

Región:
España

Teléfono:
+34661378805

Tarjeta de crédito:
1111111111111111

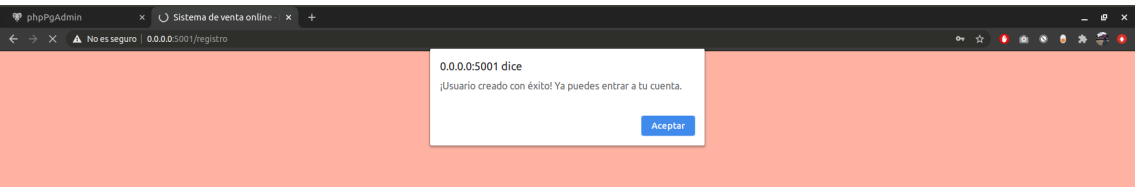
Tipo de tarjeta:
Visa

Expiración de la tarjeta:
junio de 2022

Género:
☒ Hombre ☐ Mujer

Registrar Borrar

Tu pagina web de confianza para comprar y alquilar DVDs - Web realizada por Luis Miguel y Adrian para SE



Como podemos observar, el usuario se ha creado correctamente. Para comprobarlo, nos conectamos a la base de datos y buscamos nuestro nuevo usuario.

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 Siguiente Fin >>

Acciones	customerid	firstname	lastname	address1	address2	city	state	zip	country	region	email	phone	creditcardtype	creditcard	creditcardexpiration	username
Editar Eliminar	14094	Adrian	San Felipe	Capitan Francisco Sanchez N34 3E		Alcobendas	Madrid	28100	España	España	adrianfe1@gmail.com	+34661378805	Visa	1111111111111111	2022-06	adrian

Una vez encontrado, podemos comprobar que podemos iniciar sesión con él en la página web:

Al introducir los datos, comprobamos que en efecto inicia sesión correctamente.

Sistema de venta online

Log Out

Historial de usuario

Catalogo

5 usuarios

Historial de adrian

Fecha	Importe	Detalles
-------	---------	----------

Cambiar saldo: Aceptar Carrito