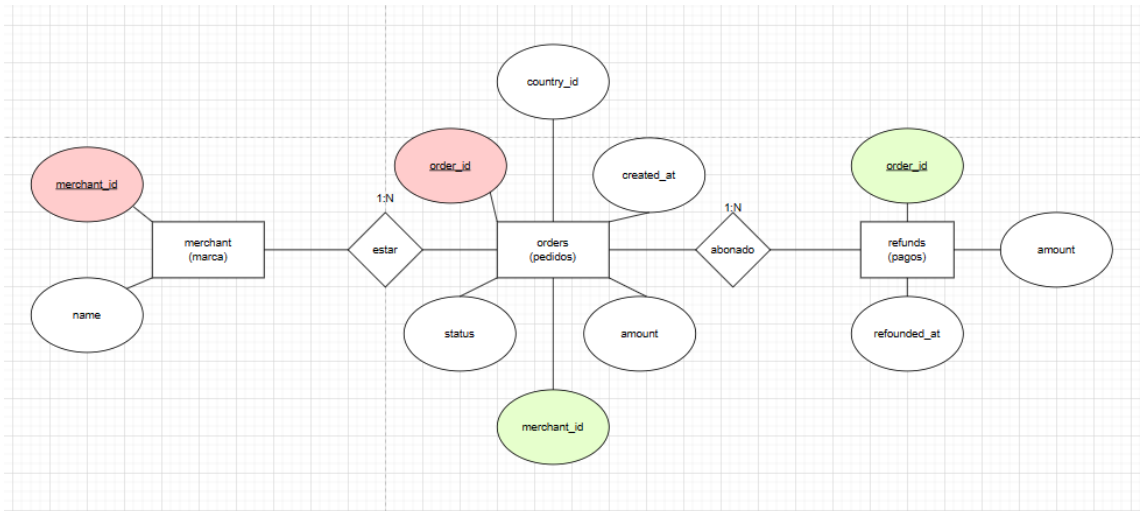


Bases de Datos con SQL: TAREA FINAL

PARTE 1

1.1 Modelo entidad-relacion. DRAW.io



[Diagrama sin título.drawio - draw.io \(diagrams.net\)](#)

1.2 Texto con las consultas de creación de la base de datos y el esquema ERR

En primer lugar, hay que aclarar que el método que he utilizado ha sido con la funcionalidad reverse engineering, a diferencia del aprendizaje en clase (forward engineering). De esta manera he cargado las tablas con el data import wizard. Una vez cargadas y configuradas para cada dato, se ha procedido a establecer las claves primarias en caso de tenerlas y las claves foráneas a través de las cuales relacionaremos las tablas (Pk_tabla1→fk_PK_tabla1)

Victor_Moro_Ludeña* orders SQL File 11* SQL File 12* merchants - Table orders - Table x

Table Name: orders Schema: prestamos_ucm

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
order_id	VARCHAR(40)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
created_at	DATETIME	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
status	VARCHAR(40)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
amount	DOUBLE	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
merchant_id	VARCHAR(40)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
country	VARCHAR(40)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL

Column Name: Data Type:

Charset/Collation: Default:

Comments:

Storage: ☐ Virtual ☐ Stored

☐ Primary Key ☐ Not Null ☐ Unique

☐ Binary ☐ Unsigned ☐ Zero Fill

☐ Auto Increment ☐ Generated

Table Name: Schema: **prestamos_ucm**

Charset/Collation: Engine:

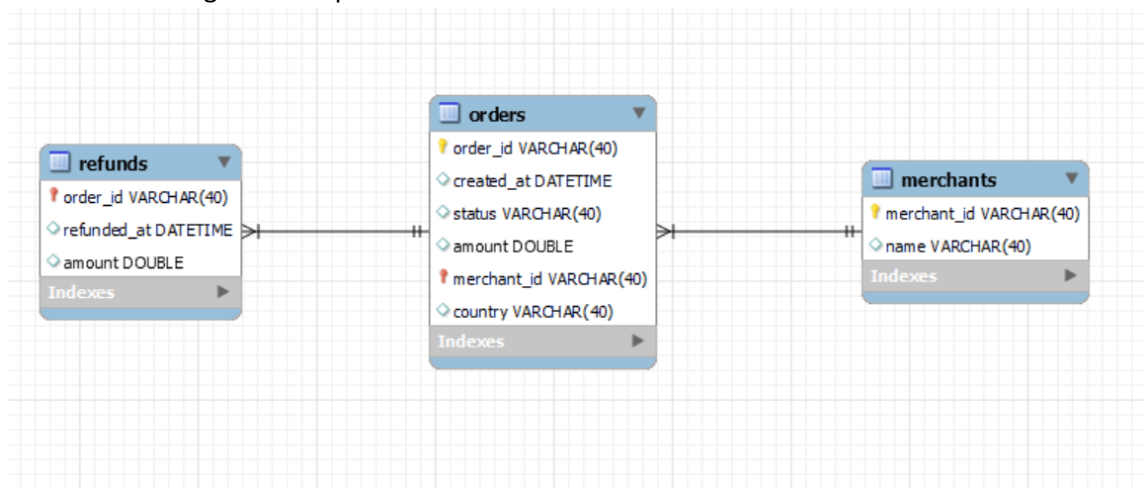
Comments:

Foreign Key Name	Referenced Table	Column	Referenced Column	Foreign Key Options
fk_merchant_id_merchants	prestamos_ucm`.`merchants`			On Update: <input type="text"/> On Delete: <input type="text"/>

Foreign Key Comment generation

Una vez tenemos las tablas cargadas y configuradas desde el localhost, podemos seleccionar el esquema que hemos creado con nuestras tablas y con la acción Database → reverse engineer, hemos conseguido hacer el esquema con las relaciones ya establecidas.

Obteniendo el siguiente esquema:



Con el siguiente código de creación para la base de datos:

-- MySQL Workbench Forward Engineering

```

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';
  
```

-- Schema mydb

-- Schema prestamos_ucm

```
-----  
-----  
-- Schema prestamos_ucm  
-----
```

```
CREATE SCHEMA IF NOT EXISTS `prestamos_ucm` DEFAULT CHARACTER SET utf8mb4  
COLLATE utf8mb4_0900_ai_ci ;  
USE `prestamos_ucm` ;
```

```
-----  
-- Table `prestamos_ucm`.`merchants`  
-----
```

```
CREATE TABLE IF NOT EXISTS `prestamos_ucm`.`merchants` (  
  `merchant_id` VARCHAR(40) NOT NULL,  
  `name` VARCHAR(40) NULL DEFAULT NULL,  
  PRIMARY KEY (`merchant_id`))  
ENGINE = InnoDB  
DEFAULT CHARACTER SET = utf8mb4  
COLLATE = utf8mb4_0900_ai_ci;
```

```
-----  
-- Table `prestamos_ucm`.`orders`  
-----
```

```
CREATE TABLE IF NOT EXISTS `prestamos_ucm`.`orders` (  
  `order_id` VARCHAR(40) NOT NULL,  
  `created_at` DATETIME NULL DEFAULT NULL,  
  `status` VARCHAR(40) NULL DEFAULT NULL,  
  `amount` DOUBLE NULL DEFAULT NULL,  
  `merchant_id` VARCHAR(40) NOT NULL DEFAULT NULL,  
  `country` VARCHAR(40) NULL DEFAULT NULL,  
  PRIMARY KEY (`order_id`, `merchant_id`),  
  INDEX `fk_merchant_id_merchants_idx` (`merchant_id` ASC) VISIBLE,  
  CONSTRAINT `fk_merchant_id_merchants`  
    FOREIGN KEY (`merchant_id`)  
      REFERENCES `prestamos_ucm`.`merchants` (`merchant_id`))  
ENGINE = InnoDB  
DEFAULT CHARACTER SET = utf8mb4  
COLLATE = utf8mb4_0900_ai_ci;
```

```
-----  
-- Table `prestamos_ucm`.`refunds`  
-----
```

```
CREATE TABLE IF NOT EXISTS `prestamos_ucm`.`refunds` (  
  `order_id` VARCHAR(40) NOT NULL DEFAULT NULL,  
  `refunded_at` DATETIME NULL DEFAULT NULL,  
  `amount` DOUBLE NULL DEFAULT NULL,
```

```
INDEX `fk_order_id_order_idx` (`order_id` ASC) VISIBLE,  
PRIMARY KEY (`order_id`),  
CONSTRAINT `fk_order_id_order`  
FOREIGN KEY (`order_id`)  
REFERENCES `prestamos_ucm`.`orders` (`order_id`))  
ENGINE = InnoDB
```

```
DEFAULT CHARACTER SET = utf8mb4  
COLLATE = utf8mb4_0900_ai_ci;
```

```
SET SQL_MODE=@OLD_SQL_MODE;  
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;  
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

PARTE 2

A partir de las tablas incluidas en la base de datos tarea_ucm, vamos a realizar las siguientes consultas:

- 1- Realizamos una consulta donde obtengamos por país y estado de operación, el total de operaciones y su importe promedio. La consulta debe cumplir las siguientes condiciones:
 - a. Operaciones posteriores al 01-07-2015
 - b. Operaciones realizadas en Francia, Portugal y España.
 - c. Operaciones con un valor mayor de 100 € y menor de 1500€ Ordenamos los resultados por el promedio del importe de manera descendente.

Código:

```
SELECT DISTINCT country,  
status,  
COUNT(*) AS num_operaciones,  
AVG(amount) AS promedio_amount  
FROM orders  
WHERE created_at > '2015-07-01'  
AND country IN ('Francia', 'Portugal', 'España')  
AND amount BETWEEN 100 AND 1500  
GROUP BY country, status  
ORDER BY promedio_amount DESC;
```

SQL File 11*

```

3 SELECT DISTINCT country,
4 status,
5 COUNT(*) AS num_operaciones,
6 AVG(amount) AS promedio_amount
7 FROM orders
8 WHERE created_at > '2015-07-01'
9 AND country IN ('Francia', 'Portugal', 'España')
10 AND amount BETWEEN 100 AND 1500
11 GROUP BY country, status
12 ORDER BY promedio_amount DESC;

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: I A

	country	status	num_operaciones	promedio_amount
▶	Portugal	CLOSED	7	522.3814285714286
	España	DELINQUENT	21	423.4757142857143
	España	ACTIVE	172	410.6729651162792
	Portugal	ACTIVE	5	392.97999999999996
	Francia	CLOSED	44	391.2227272727272

Result 15 x

2- Realizamos una consulta donde obtengamos los 3 países con el mayor número de operaciones, el total de operaciones, la operación con un valor máximo y la operación con el valor mínimo para cada país. La consulta debe cumplir las siguientes condiciones: a. Excluimos aquellas operaciones con el estado “Delinquent” y “Cancelled” b. Operaciones con un valor mayor de 100 €

Código:

```

SELECT DISTINCT country ,
count(country IN (country)) as n_total_por_pais,
MAX(amount) AS mayor_cantidad,
MIN(amount) AS menor_cantidad
####MAX(CASE WHEN amount = (SELECT MAX(amount) FROM orders) THEN order_id END)
AS order_id_maximo_amount
####(SELECT order_id
#FROM orders
#WHERE amount=max(amount))
####He intentado sacar el order_id asociado al valor de amount(max y min) para cada
linea, pero no
#he conseguido hacerlo con estos códigos de arriba, a ver si hay alguna forma mas facil o
# soy yo que me he querido complicar:(
FROM orders
WHERE status NOT IN ('CANCELLED', 'DELINQUENT') AND amount>100
GROUP BY country
ORDER BY n_total_por_pais DESC
LIMIT 3;

```

```

13
14 #apartado 2
15 • SELECT DISTINCT country ,
16     count(country IN (country)) as n_total_por_pais,
17     MAX(amount) AS mayor_cantidad,
18     MIN(amount) AS menor_cantidad
19     ###MAX(CASE WHEN amount = (SELECT MAX(amount) FROM orders) THEN order_id END) AS order_id_maximo
20     ###(SELECT order_id
21     #FROM orders
22     #WHERE amount=max(amount))

```

country	n_total_por_pais	mayor_cantidad	menor_cantidad
España	342	2960.87	101
Francia	141	1863.98	100.88
Italia	75	1299	107.99

PARTE 3.

A partir de las tablas incluidas en la base de datos tarea_ucm vamos a realizar las siguientes consultas:

- 1- . Realizamos una consulta donde obtengamos, por país y comercio, el total de operaciones, su valor promedio y el total de devoluciones. La consulta debe cumplir las siguientes condiciones:
 - a. Se debe mostrar el nombre y el id del comercio.
 - b. Comercios con más de 10 ventas.
 - c. Comercios de Marruecos, Italia, España y Portugal.
 - d. Creamos un campo que identifique si el comercio acepta o no devoluciones. Si no acepta (total de devoluciones es igual a cero) el campo debe contener el valor “No” y si sí lo acepta (total de devoluciones es mayor que cero) el campo debe contener el valor “Sí”. Llamaremos al campo “acepta_devoluciones”.
- Ordenamos los resultados por el total de operaciones de manera ascendente.

Código:

```

SELECT M.merchant_id AS id_comercio,
M.name AS marca,
O.country as pais,
count(DISTINCT O.order_id) as operaciones_pais_marca,
ROUND(AVG(O.amount),3) AS valor_promedio,
count(DISTINCT R.order_id) as total_devoluciones,
IF(SUM(R.amount)>0,'Si','No') AS aceptacion_devolucion
FROM orders as O
INNER JOIN merchants as M ON O.merchant_id=M.merchant_id
LEFT JOIN refunds as R ON O.order_id=R.order_id
WHERE country IN ('España','Marruecos','Italia','Portugal')
GROUP BY id_comercio, marca, pais
HAVING operaciones_pais_marca > 10
ORDER BY operaciones_pais_marca ASC;

```

tareaSQL_Victor_Moro_Ludeña x orders SQL File 11*

Limit to 50000 rows

```

32 #EJERCICIO 3
33 #apartado 1
34
35 • SELECT M.merchant_id AS id_comercio,
36 M.name AS marca,
37 O.country as pais,
38 count(DISTINCT O.order_id) as operaciones_pais_marca,
39 ROUND(AVG(O.amount),3) AS valor_promedio,
40 count(DISTINCT R.order_id) as total_devoluciones,
41 IF(SUM(R.amount)>0,'Si','No') AS aceptacion_devolucion

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: [IA](#)

	id_comercio	marca	pais	operaciones_pais_marca	valor_promedio	total_devoluciones	aceptacion
▶	pk_743f2fdec876b75e975c005	Pepe Jeans	España	11	171.994	0	No
	pk_317b4fc6fd80a5f8fb2ff216	Calcedonia	Marruecos	12	365.357	2	Si
	pk_736c7094ea96eda38b098f56	Massimo Dutti	España	13	169.877	0	No
	pk_c15afcbd3a31b732f097ba7b	Havainas	España	16	323.021	0	No

2- Realizamos una consulta donde vamos a traer todos los campos de las tablas operaciones y comercios. De la tabla devoluciones vamos a traer el conteo de devoluciones por operación y la suma del valor de las devoluciones. Una vez tengamos la consulta anterior, creamos una vista con el nombre orders_view dentro del esquema tarea_ucm con esta consulta.

Nota: La tabla refunds contiene más de una devolución por operación por lo que, para hacer el cruce, es muy importante que agrupemos las devoluciones.

Código:

```

CREATE VIEW tarea_ucm.orders_view AS
SELECT
o.*,
m.name AS MARCA,
COUNT(DISTINCT r.order_id) AS devoluciones,
SUM(r.amount) AS suma_devoluciones
FROM orders o
INNER JOIN merchants m ON o.merchant_id = m.merchant_id
LEFT JOIN refunds r ON o.order_id = r.order_id
GROUP BY o.order_id, m.name;

```


The screenshot shows a SQL IDE window with a query editor and a result grid. The query editor contains the following SQL code:

```

50
51 #apartado 2
52 CREATE VIEW tarea_ucm.orders_view AS
53 SELECT
54 o.*,
55 m.name AS MARCA,
56 COUNT(DISTINCT r.order_id) AS devoluciones,
57 SUM(r.amount) AS suma_devoluciones
58 FROM orders o
59 INNER JOIN merchants m ON o.merchant_id = m.merchant_id
60 LEFT JOIN refunds r ON o.order_id = r.order_id
61 GROUP BY o.order_id, m.name;

```

The result grid displays the following data:

order_id	created_at	status	amount	merchant_id	country	MARCA	devoluc
5c370ef03e78fa6aacf5f623	2015-12-15 08:21:21	ACTIVE	726.75	pk_c447a91e755425d163df6837	España	YouTube ...	1
5c3ef3f70aee697c1ba7e93a	2015-12-14 16:34:53	ACTIVE	423.43	pk_c15afcbd3a31b732f097ba7b	Francia	Havainas	1
5c3ef3f70aee697c1ba7e93b	2015-12-15 09:11:09	ACTIVE	137.37	pk_c15afcbd3a31b732f097ba7b	Marruecos	Havainas	1
5c3ef6460aee697c1ba82bcd	2015-12-15 07:31:29	ACTIVE	124.46	pk_e1af716f8d336aceb6237bf5	Italia	Kindle	1

PARTE 4.

Para la realización de este ejercicio se ha realizado una consulta para conseguir la tasa de devoluciones en porcentaje por país y marca(comercio), de esta manera podemos analizar aspectos como la calidad de las ventas, la satisfacción del cliente con el servicio y con el producto y en como difiere el servicio de venta en cada país comparándolo con la tasa de devoluciones calculada. Además, se hace un filtro en la consulta para filtrar solo donde el total de ventas sea mayor que 0, pudiendo analizar así la base de datos y solo obtener los datos que van teniendo ventas. Es decir, que se vaya actualizando cada vez que se consulte y vayan apareciendo tasas de devolución en función de si hay ventas o no, ya que no tiene sentido hablar de devolución si no ha habido una compra antes.

Código:

```

SELECT
M.name AS nombre_comercio,
O.country AS pais,
COUNT(DISTINCT O.order_id) AS total_ventas,
COUNT(DISTINCT R.order_id) AS total_devoluciones,
ROUND((COUNT(DISTINCT R.order_id) / NULLIF(COUNT(DISTINCT O.order_id), 0)) * 100, 2)
AS tasa_devoluciones_porcentaje
FROM merchants M
LEFT JOIN orders O ON M.merchant_id = O.merchant_id
LEFT JOIN refunds R ON O.order_id = R.order_id
GROUP BY M.name, pais
HAVING total_ventas >0
ORDER BY pais, tasa_devoluciones_porcentaje DESC;

```

tareaSQL_Victor_Moro_Luderia*

orders

SQL File 11*

Limit to 50000 rows

```

66 SELECT
67     M.name AS nombre_comercio,
68     O.country AS pais,
69     COUNT(DISTINCT O.order_id) AS total_ventas,
70     COUNT(DISTINCT R.order_id) AS total_devoluciones,
71     ROUND((COUNT(DISTINCT R.order_id) / NULLIF(COUNT(DISTINCT O.order_id), 0)) * 100, 2) AS tasa_devoluciones_porcentaje
72 FROM merchants M
73 LEFT JOIN orders O ON M.merchant_id = O.merchant_id
74 LEFT JOIN refunds R ON O.order_id = R.order_id
75 GROUP BY M.name, pais
76 HAVING total_ventas >0
77 ORDER BY pais, tasa_devoluciones_porcentaje DESC;

```

Result Grid

Filter Rows:

Export:

Wrap Cell Content:

	nombre_comercio	pais	total_ventas	total_devoluciones	tasa_devoluciones_porcentaje
▶	Apple music	Francia	3	1	33.33
	Havainas	Marruecos	4	1	25.00
	Calcedonia	Portugal	5	1	20.00
	Calcedonia	Marruecos	12	2	16.67
	Kindle	Italia	7	1	14.29

Result 26

Lo que podemos ver de este output, es que Apple music es la marca con mas devoluciones a nivel global, podemos decir que el modelo de negocio que tienen, en el que actualizan sus hardwares cada año, puede hacer que inviertan menos en producción precisa y salga con más defectos, este dato supone que 1 de cada 3 dispositivos o compras de la marca es devuelta por el cliente.