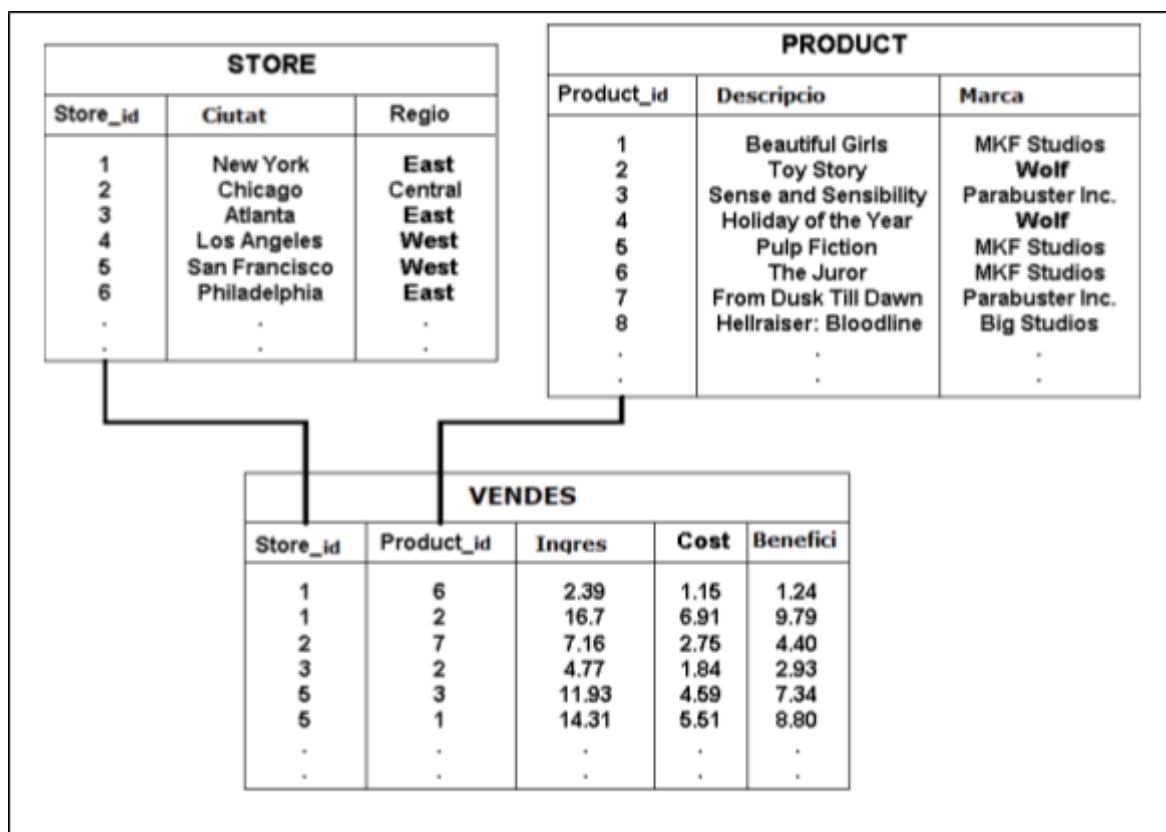


Pràctica RA4-DDL CREATE TABLE-CREATE VIEW

Exercici 1 - taules

Agafant la base de dades de "botiga" com exemple crear un nou esquema anomenat "**store**" on hi haurà les taules necessàries per servir de repositori de les dades que es mostra en la següent imatge/diagrama:

- hi ha botigues
- hi ha productes
- hi ha un inventari de vendes, donada una botiga i un producte:
 - quants ingressos ha tingut la botiga de la venda d'aquest producte, *sense indicar-ne la quantitat venuda*.
 - quina ha estat el cost d'aquest producte per la botiga.
 - de tot plegat es calcula el benefici obtingut de cada producte.



Les tasques a fer són:

1. Identificar la relació de taules a crear en aquest nou esquema "**store**".
 - *A banda de les taules que es dedueixen de la imatge cal identificar altres possibles taules que es podrien crear per suportar el model, per ex: una taula de ciutats, una de marques....*
2. Identificar les columnes que ha de tenir cada taula de BD i de quin tipus.
 - *No hi ha una relació directe entre la relació de columnes que es mostra en la imatge i la relació de columnes que ha de tenir les taules que creeu.*
3. Identificar la clau primària de cada taula.
4. Identificar les relacions d'integritat referencial entre les taules.
5. Identificar altres possibles restriccions (constraints) que es podrien aplicar a les columnes de les taules.
 - Valors null , not null.
 - Checks

Una vegada has recollit tota la informació per a crear les taules de la BD. Construeix les sentències de CREATE TABLE corresponents i executa-les.

A l'informe de la pràctica s'han de mostrar les sentències de CREATE TABLE que s'han fet, cal que estiguin en format text copiable, i explicar una mica per a que serveix cada taula i les relacions que s'han definit entre elles.

***/ EJERCICIO 1**

```
CREATE SCHEMA store;  
/*CREATE SCHEMA store      1 row(s) affected      0.015 sec
```

***/ Creamos el esquema**

```
CREATE TABLE store.stores (  
    store_id INT AUTO_INCREMENT PRIMARY KEY,  
    store_name VARCHAR(100) NOT NULL,  
    city VARCHAR(100),  
    region VARCHAR(100));
```

```
/* CREATE TABLE store.stores (store_id INT AUTO_INCREMENT PRIMARY KEY,store_name  
VARCHAR(100) NOT NULL,city VARCHAR(100),region VARCHAR(100))0 row(s) affected    0.031 sec
```

/* Es una lista de todas las tiendas.Cada tienda tiene un identificador único store_id. También guarda información básica sobre la tienda, como su nombre, la ciudad donde se encuentra y la región.

```
CREATE TABLE store.products (  
    product_id INT AUTO_INCREMENT PRIMARY KEY,  
    product_name VARCHAR(100) NOT NULL,  
    product_brand VARCHAR(100),  
    cost_price DECIMAL(10, 2),  
    selling_price DECIMAL(10, 2));
```

```
/* CREATE TABLE store.products (product_id INT AUTO_INCREMENT PRIMARY KEY,product_name  
VARCHAR(100) NOT NULL,product_brand VARCHAR(100),cost_price DECIMAL(10, 2),selling_price  
DECIMAL(10, 2))      0 row(s) affected      0.031 sec
```

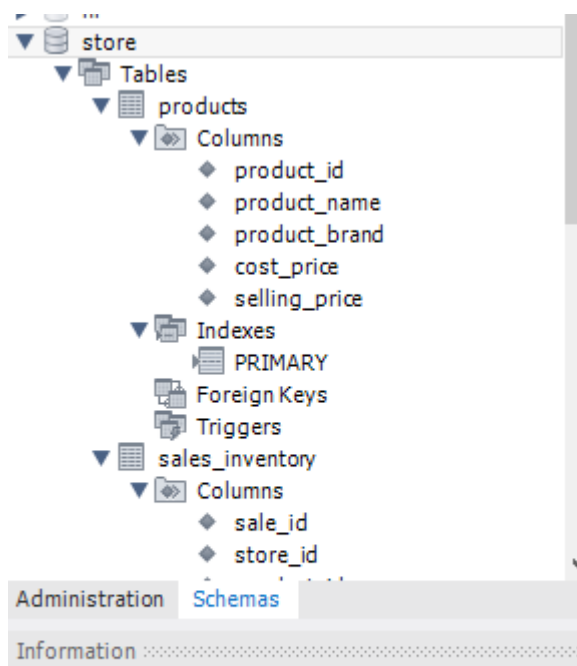
/* Es una lista de los productos disponibles.Cada producto tiene un identificador único product_id. Incluye datos como, El nombre del producto, La marca, Coste, Precio venta.

```
CREATE TABLE store.sales_inventory (  
    sale_id INT AUTO_INCREMENT PRIMARY KEY,  
    store_id INT NOT NULL,  
    product_id INT NOT NULL,  
    revenue DECIMAL(10, 2),  
    cost DECIMAL(10, 2),  
    profit DECIMAL(10, 2) GENERATED ALWAYS AS (revenue - cost) STORED, -- Columna calculada  
    FOREIGN KEY (store_id) REFERENCES store.stores(store_id),  
    FOREIGN KEY (product_id) REFERENCES store.products(product_id));
```

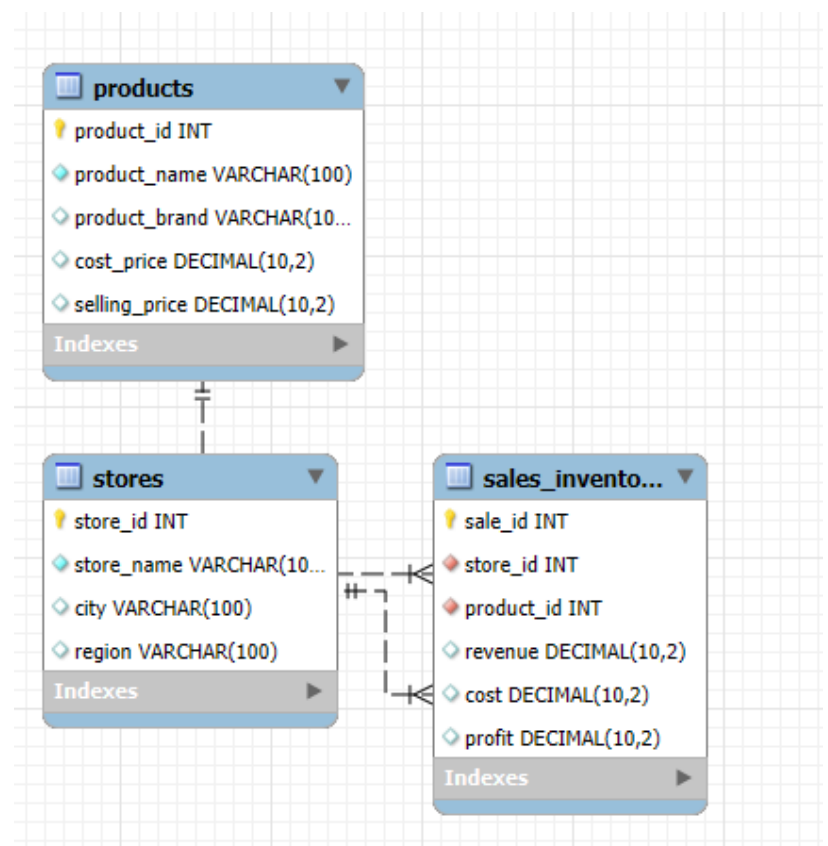
```
/* CREATE TABLE store.sales_inventory (sale_id INT AUTO_INCREMENT PRIMARY KEY,store_id INT  
NOT NULL,product_id INT NOT NULL,revenue DECIMAL(10, 2),cost DECIMAL(10,2), profit  
DECIMAL(10, 2) GENERATED ALWAYS AS (revenue - cost) STORED,FOREIGN KEY (store_id)  
REFERENCES store.stores(store_id),FOREIGN KEY (product_id) REFERENCES  
store.products(product_id))0 row(s) affected      0.031 sec
```

/* Es un registro de las ventas realizadas, Relaciona qué tienda "store_id" vendió qué producto "product_id", los ingresos generados por la venta, El coste del producto para la tienda, El beneficio obtenido "profit", que es la diferencia entre los ingresos y el coste.

Action	Message	Duration / Fetch
CREATE SCHEMA store	1 row(s) affected	0.015 sec
CREATE TABLE store.stores (store_id INT A...	0 row(s) affected	0.031 sec
CREATE TABLE store.products (product_id l...	0 row(s) affected	0.031 sec
CREATE TABLE store.sales_inventory (sale_j...	0 row(s) affected	0.031 sec

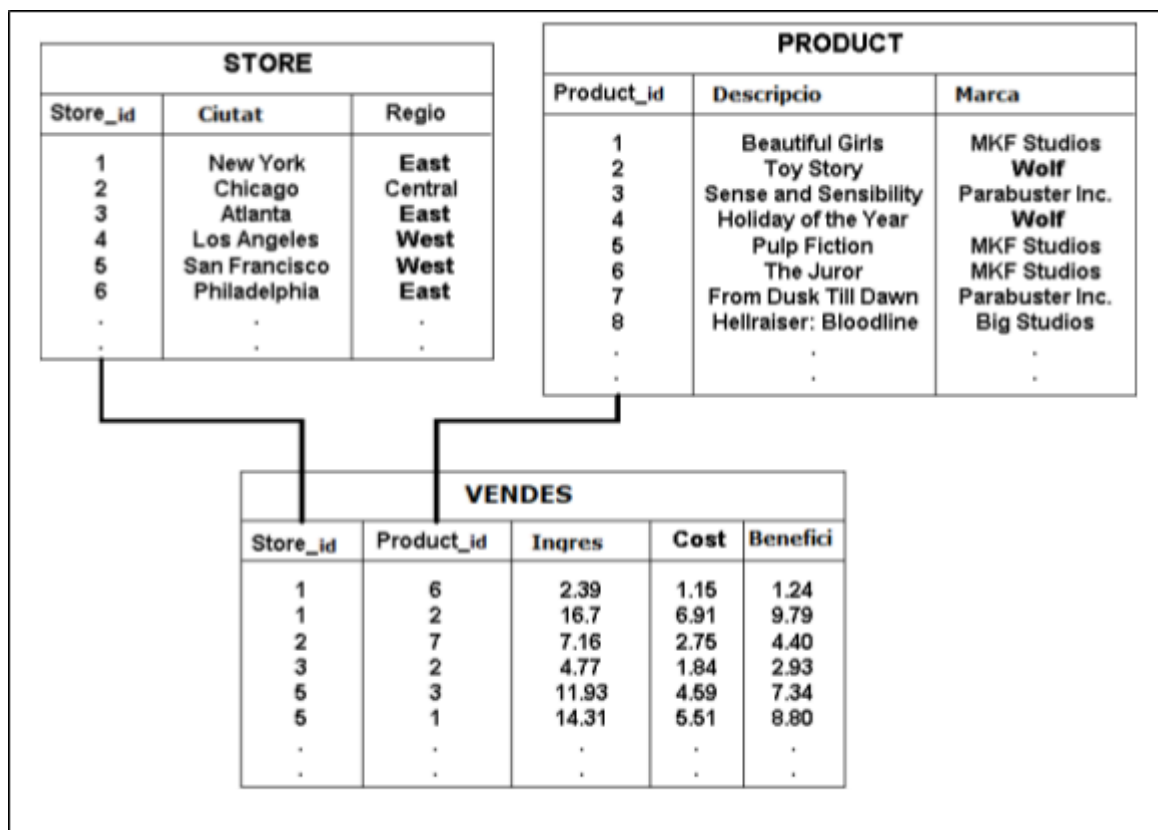


Schema: store



Exercici 2 - taules

Fent servir les noves taules creades a la BD store, inserir-hi informació per a poder-hi fer consultes. Les dades a inserir no han de ser les mateixes que es mostren a la imatge, poden ser altres ciutat, regions



Les tasques a fer són les queries per verificar les dades introduïdes:

1. Select "Store" que retorni la llista de botigues
2. Select "Product" que retorni la llista de productes
3. Select "Vendes" que retorni l'inventari de vendes de cada producte a cada botiga

A l'informe de la pràctica s'han de recollir les sentències SELECT i el resultat, en format text copiable la query i com a taula, o imatge, el resultat.

INTRODUCIMOS DATOS FICTICIOS

INSERT INTO store.stores (store_name, city, region) VALUES ('Tienda Central', 'Chicago', 'Central'), ('Tienda Norte', 'New York', 'Norte'), ('Tienda Sur', 'Atlanta', 'Sur');

INSERT INTO store.stores (store_name, city, regi...	3 row(s) affected Records: 3 Duplicates: 0 War...	0,015 sec
---	---	-----------

INSERT INTO store.products (product_name, product_brand, cost_price, selling_price) VALUES ('Producto A', 'Marca X', 10.50, 15.00), ('Producto B', 'Marca Y', 5.25, 8.00), ('Producto C', 'Marca Z', 20.00, 30.00);

INSERT INTO store.products (product_name, pr...	3 row(s) affected Records: 3 Duplicates: 0 War...	0,000 sec
---	---	-----------

INSERT INTO store.products (product_name, product_brand, cost_price, selling_price) VALUES ('Producto A', 'Marca X', 10.50, 15.00), ('Producto B', 'Marca Y', 5.25, 8.00), ('Producto C', 'Marca Z', 20.00, 30.00);






INSERT INTO store.products (product_name, pr...	3 row(s) affected Records: 3 Duplicates: 0 War...	0,000 sec
---	---	-----------

INSERT INTO store.products (product_name, product_brand, cost_price, selling_price) VALUES ('Producto A', 'Marca X', 10.50, 15.00), ('Producto B', 'Marca Y', 5.25, 8.00), ('Producto C', 'Marca Z', 20.00, 30.00);

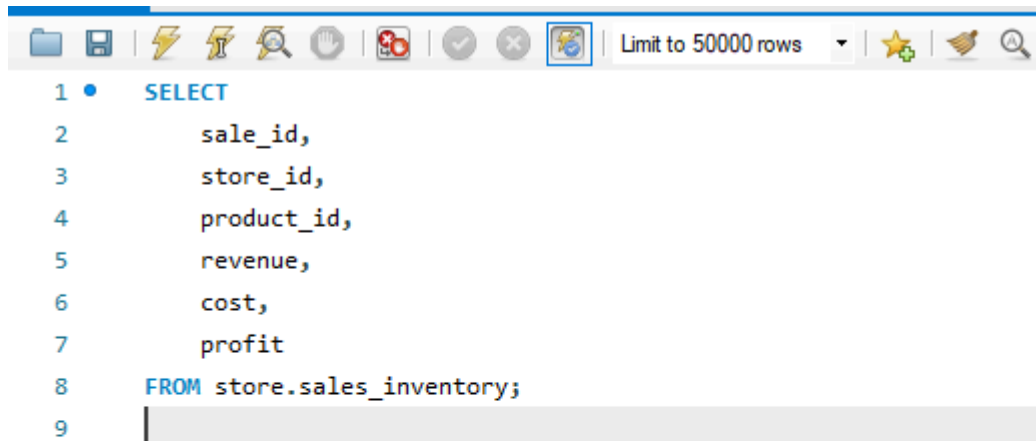
INSERT INTO store.products (product_name, pr...	3 row(s) affected Records: 3 Duplicates: 0 War...	0,000 sec
---	---	-----------

INSERT INTO store.sales_inventory (store_id, product_id, revenue, cost) VALUES (1, 1, 150.00, 105.00), -- Tienda Central vendió Producto A (1, 2, 80.00, 52.50), -- Tienda Central vendió Producto B (2, 3, 300.00, 200.00), -- Tienda Norte vendió Producto C (3, 1, 45.00, 31.50); -- Tienda Sur vendió Producto A

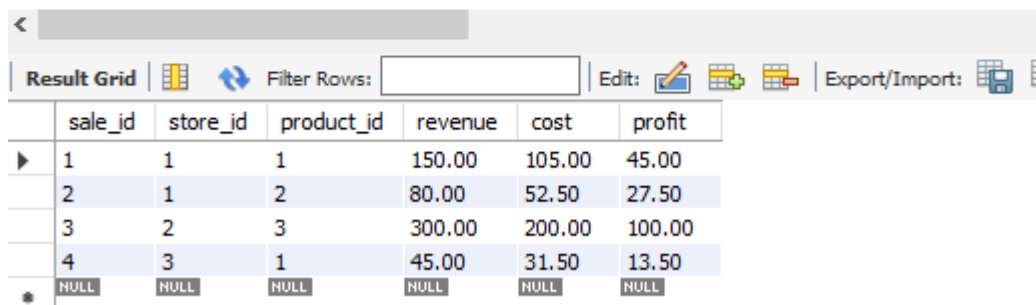
INSERT INTO store.sales_inventory (store_id, pr...	4 row(s) affected Records: 4 Duplicates: 0 War...	0,000 sec
--	---	-----------

Result Grid					
Filter Rows: <input type="text"/>					
Edit:   					
Export/Import:  					
Wrap Cell Cont					
	product_id	product_name	product_brand	cost_price	selling_price
1	1	Producto A	Marca X	10.50	15.00
2	2	Producto B	Marca Y	5.25	8.00
3	3	Producto C	Marca Z	20.00	30.00
4	4	Producto A	Marca X	10.50	15.00
5	5	Producto B	Marca Y	5.25	8.00
6	6	Producto C	Marca Z	20.00	30.00
7	7	Producto A	Marca X	10.50	15.00
8	8	Producto B	Marca Y	5.25	8.00
9	9	Producto C	Marca Z	20.00	30.00
10	NULL	NULL	NULL	NULL	NULL

```
SELECT
    sale_id,
    store_id,
    product_id,
    revenue,
    cost,
    profit
FROM store.sales_inventory;
```



```
1 • SELECT
2     sale_id,
3     store_id,
4     product_id,
5     revenue,
6     cost,
7     profit
8 FROM store.sales_inventory;
9
```



	sale_id	store_id	product_id	revenue	cost	profit
▶	1	1	1	150.00	105.00	45.00
	2	1	2	80.00	52.50	27.50
	3	2	3	300.00	200.00	100.00
	4	3	1	45.00	31.50	13.50
*	NULL	NULL	NULL	NULL	NULL	NULL

INSERT INTO store.stores (store_name, city, r...	3 row(s) affected Records: 3 Duplicates: 0 ...	0,015 sec
INSERT INTO store.products (product_name,...	3 row(s) affected Records: 3 Duplicates: 0 ...	0,000 sec
INSERT INTO store.products (product_name,...	3 row(s) affected Records: 3 Duplicates: 0 ...	0,000 sec
INSERT INTO store.products (product_name,...	3 row(s) affected Records: 3 Duplicates: 0 ...	0,000 sec
INSERT INTO store.sales_inventory (store_id,...	4 row(s) affected Records: 4 Duplicates: 0 ...	0,000 sec
SELECT * FROM store.stores LIMIT 0, 50000	3 row(s) returned	0,000 sec / 0,000 sec
SELECT * FROM store.products LIMIT 0, 500...	9 row(s) returned	0,000 sec / 0,000 sec
SELECT sale_id, store_id, product_i...	4 row(s) returned	0,000 sec / 0,000 sec

Exercici 3 - taules

Fes consultes al diccionari de dades, "INFORMATION_SCHEMA", per obtenir informació de la definició de les taules que acabes de crear.

- informació referent a taules
- informació referent a columnes
- informació referent a constraints

Escriu 5 sentències de consultes al diccionari de dades, s'han d'incloure a l'informe de la pràctica la consulta i el resultat, en format text copiable la query i com a taula, o imatge, el resultat.

```
SELECT TABLE_NAME FROM INFORMATION_SCHEMA.TABLES WHERE  
TABLE_SCHEMA = 'store';
```

SELECT TABLE_NAME FROM INFORMATI... 3 row(s) returned 0,016 sec / 0,000 sec

Result Grid	Filter Rows:	Export:
TABLE_NAME		
products		
sales_inventory		
stores		

```
SELECT COLUMN_NAME, DATA_TYPE, COLUMN_TYPE  
FROM INFORMATION_SCHEMA.COLUMNS  
WHERE TABLE_SCHEMA = 'store' AND TABLE_NAME = 'sales_inventory';
```

	COLUMN_NAME	DATA_TYPE	COLUMN_TYPE
▶	sale_id	int	int
	store_id	int	int
	product_id	int	int
	revenue	decimal	decimal(10,2)
	cost	decimal	decimal(10,2)
	profit	decimal	decimal(10,2)

SELECT COLUMN_NAME, DATA_TYPE, CO... 6 row(s) returned 0,000 sec / 0,000 sec

```
SELECT TABLE_NAME, COLUMN_NAME FROM  
INFORMATION_SCHEMA.KEY_COLUMN_USAGE WHERE TABLE_SCHEMA =  
'store' AND CONSTRAINT_NAME = 'PRIMARY';
```

	TABLE_NAME	COLUMN_NAME
▶	products	product_id
	sales_inventory	sale_id
	stores	store_id

SELECT TABLE_NAME, COLUMN_NAME F...	3 row(s) returned	0,000 sec / 0,000 sec
-------------------------------------	-------------------	-----------------------

```
SELECT TABLE_NAME, COLUMN_NAME, REFERENCED_TABLE_NAME,  
REFERENCED_COLUMN_NAME  
FROM INFORMATION_SCHEMA.KEY_COLUMN_USAGE  
WHERE TABLE_SCHEMA = 'store' AND REFERENCED_TABLE_NAME IS NOT  
NULL;
```

SELECT TABLE_NAME, COLUMN_NAME, R...	2 row(s) returned	0,015 sec / 0,000 sec
--------------------------------------	-------------------	-----------------------

	TABLE_NAME	COLUMN_NAME	REFERENCED_TABLE_NAME	REFERENCED_COLUMN_NAME
▶	sales_inventory	store_id	stores	store_id
	sales_inventory	product_id	products	product_id

```
SELECT CONSTRAINT_NAME, CONSTRAINT_TYPE  
FROM INFORMATION_SCHEMA.TABLE_CONSTRAINTS  
WHERE TABLE_SCHEMA = 'store' AND TABLE_NAME = 'sales_inventory';
```

	CONSTRAINT_NAME	CONSTRAINT_TYPE
▶	PRIMARY	PRIMARY KEY
	sales_inventory_ibfk_1	FOREIGN KEY
	sales_inventory_ibfk_2	FOREIGN KEY

SELECT CONSTRAINT_NAME, CONSTRAI...	3 row(s) returned	0,016 sec / 0,000 sec
-------------------------------------	-------------------	-----------------------

Exercici 4 - vistes

Fent servir les noves taules creades a l'esquema store, crear una **VISTA** per poder fer un LLISTAT DE VENDES semblant al que es mostra a la imatge:

LLISTAT VENDES				
Ciutat	Descripció	Ingres	Cost	Benefici
New York	Toy Story	2.39	1.15	1.24
New York	Pulp Fiction	16.7	6.91	9.79
Chicago	Toy Story	7.16	2.75	4.40
Chicago	Pulp Fiction	4.77	1.84	2.93
Chicago	The Juror	11.93	4.59	7.34
Los Angeles	Toy Story	14.31	5.51	8.80
-	-	-	-	-
-	-	-	-	-

La vista ha de tenir la llista de columnes necessària per a poder fer consultes del tipus:

- productes d'una ciutat
- productes d'una regió
- ...

El primer pas a l'hora de crear la vista és transformar l'inventari de vendes, on hi ha codis per la botiga i el producte, en una inventari on en lloc de codis hi ha noms.

Les tasques a fer són:

- Crear la vista.
- A partir de la vista fer les següents queries:
 1. Llistat de productes de "Chicago"
 2. Total d'ingressos, costos, beneficis de "Chicago"
 3. Llistat de productes de la regió "Central"
 4. Total d'ingressos, costos, beneficis de la regió "Central"
 5. Total de productes de cada ciutat.
 6. El producte amb ingressos més elevats de cada ciutat.

Si heu fet servir uns noms de ciutat i regions diferents feu servir els vostres.

A l'informe de la pràctica s'ha de recollir la sentència CREATE VIEW (en format text copiable).

De cada query feta a partir de la vista cal fer-ne una versió sense fer servir la vista només les taules de la BD, adjuntar evidència de que les dues queries donen el mateix resultat, les queries en format text copiable i el resultat com a taula o imatge.

```
CREATE VIEW store.sales_view AS SELECT s.store_name, s.city, s.region, p.product_name, p.product_brand, si.revenue, si.cost, si.profit FROM store.sales_inventory si JOIN store.stores s ON si.store_id = s.store_id JOIN store.products p ON si.product_id = p.product_id;
```

SELECT CONSTRAINT_NAME, CONSTRAI...	3 row(s) returned	0.016 sec / 0.000 sec
-------------------------------------	-------------------	-----------------------

1- Listar productos vendidos en "Chicago":

```
SELECT product_name, product_brand  
FROM store.sales_view  
WHERE city = 'Chicago';
```

SELECT product_name, product_brand FRO... 2 row(s) returned 0,000 sec / 0,000 sec

Result Grid	Filter Rows:
product_name	product_brand
Producto A	Marca X
Producto B	Marca Y

2- Total de ingresos, costos y beneficios en "Chicago":

```
SELECT SUM(revenue) AS total_revenue, SUM(cost) AS total_cost, SUM(profit) AS  
total_profit  
FROM store.sales_view  
WHERE city = 'Chicago';
```

SELECT SUM(revenue) AS total_revenue, S... 1 row(s) returned 0,000 sec / 0,000 sec

total_revenue	total_cost	total_profit
230.00	157.50	72.50

3- Listar productos vendidos en la región "Central"

```
SELECT product_name, product_brand  
FROM store.sales_view  
WHERE region = 'Central';
```

Result Grid	Filter Rows:
product_name	product_brand
Producto A	Marca X
Producto B	Marca Y

4- Total de ingresos, costos y beneficios en la región "Central"

```
SELECT SUM(revenue) AS total_revenue, SUM(cost) AS total_cost, SUM(profit) AS  
total_profit  
FROM store.sales_view  
WHERE region = 'Central';
```

Result Grid

Filter Rows:

Export:

	total_revenue	total_cost	total_profit
▶	230.00	157.50	72.50

SELECT SUM(revenue) AS total_revenue, S...	1 row(s) returned	0,000 sec / 0,000 sec
--	-------------------	-----------------------

5- Total de productos vendidos por ciudad:

```
SELECT city, COUNT(DISTINCT product_name) AS total_products  
FROM store.sales_view  
GROUP BY city;
```

Result Grid

Filter Rows:

	city	total_products
▶	Atlanta	1
	Chicago	2
	New York	1

SELECT city, COUNT(DISTINCT product_na...	3 row(s) returned	0,000 sec / 0,000 sec
---	-------------------	-----------------------

6- Producto con ingresos más altos en cada Ciudad

```
SELECT city, product_name, MAX(revenue) AS highest_revenue  
FROM store.sales_view  
GROUP BY city, product_name  
ORDER BY city, highest_revenue DESC;
```

	city	product_name	highest_revenue
►	Atlanta	Producto A	45.00
	Chicago	Producto A	150.00
	Chicago	Producto B	80.00
	New York	Producto C	300.00

SELECT city, product_name, MAX(revenue) A...	4 row(s) returned	0,000 sec / 0,000 sec
--	-------------------	-----------------------

Lliurament

En el document caldrà definir una capçalera a on aparegui les següents dades:

- Nom complet dels 2 alumnes
- nom del mòdul: *MP02. Bases de dades*
- unitat formativa: *UF2. Llenguatge SQL*
- nucli formatiu: *DDL (Data Definition Language)*

El nom del document haurà de ser “*M2_RA4_DDL_Create_Cognom1_Cognom2.pdf*”.

Es pujarà al Moodle en format PDF i no més tard de la data límit de lliurament indicada.

Avaluació

S’avaluarà:

- El contingut de les respostes
- El disseny del document tenint en compte la facilitat de lectura i correcció.
- L’aspecte general i la correcta utilització del llenguatge.
- Es **penalitzarà** si no es respecten les indicacions de cada exercici.