DOMANDE TEORICHE

1. Per database si intende una raccolta di dati organizzati in maniera ordinata e coerente, in modo tale da permettere le operazioni CRUD

2. DBMS sta per Database Management System e consiste in un sistema di gestione per database relazionali; così da permettere la creazione, la modifica e l'interrogazione dei dati.

3. Le principali clausole di uno statement SELECT esposte in ordine di esecuzione logica sono:
   FROM: indica la tabella da interrogare
   WHERE: condizione di ricerca, verranno restituite tutte le righe per la quale la condizione di ricerca è vera. Possono essere utilizzati operatori di confronto come =, <>, >=…
   GROUP BY: consente di raggruppare i record in valori univoci della group by list, importante che i campi contenuti nella group by list siano riportati nella select. Con essa bisogna creare sempre un campo diagregazione
   HAVING: filtra i gruppi restituiti dalla group by
   SLECT: restituisce il result set, ovvero tutti i campi che noi decidiamo di visionare. Nella select si possono creare anche campi di aggregazione come: somma, conteggio e media
   ORDER BY: consente di ordinare il result set così da rendere i dati più leggibili, secondo le nostre necessità.

4. La Group By mi permette di raggruppare il campo PRODUCTID eliminando i duplicati e restituisce la media (campo aggregatore scelto da me) del UNITEPRICE che ho rinominato con un ALIAS in COSTO_MEDIO

5. OLTP: Online Transactional Processing, è un sistema che garantisce la consistenza, l'integrità e la sicurezza delle transazioni svolte in tempo reale e recupera i dati da un'unica origine. Mentre l'OLAP: Online Analytical Processing è un processo che ci permette di analizzare un grande volume di dati provenienti da più origini.

6. La differenza principale tra una JOIN e una SUBQUERY sta nel result set. La Join tra due tabelle restituisce le possibili combinazioni di valori di due insiemi. Essa si distingue in CROSS JOIN (che restituisce tutte le possibili combinazioni) INNER JOIN (restituisce solo l'insieme dei campi in comune), LEFT OUTER JOIN (restituisce tutti i valori della tabella di sinistra anche per i record che non trovano corrispondenza nella tabella di destra) e RIGHT OUTER JOIN (restituisce tutti i valori della tabella di destra anche per i record che non trovano corrispondenza nella tabella di sinistra).

Mentre le SUBQUERY sono query innestate e il risultato della query esterna dipende dal risultato della query interna.

7. DML sta per Data Manipulation Language e consente di aggiornare, eliminare e inserire dati da tabelle già esistenti, mentre DDL (Data Definition Language) fa parte sempre di un linguaggio di SQL che permette di modificare, creare o eliminare database o loro struttura.

8. Per estrapolare l'anno da un campo data posso scrivere la query seguente:
   ex.
   SELECT YEAR(OrderDate) Restituisce solo l'anno delle date inserite nel campo OrderDate
   FROM factreseller Interrogo la tabella Factreseller

9. Gli operatori AND e OR vengono comunemente utilizzati nella condizione di ricerca (WHERE), quando le condizioni sono più di una. Utilizzando AND il risultato restituisce TRUE solo nel caso le espressioni booleane sono entrambe verificate, mentre OR restituisce TRUE se una delle due condizioni sono verificate.

10. Si è possibile e prende il nome di QUERY INNESTATA SCALARE INDIPENDENTE: la query interna restituisce un valore unico che funge da input per la query esterna, essa è detta indipendente perché la query interna funziona indipendentemente da quella esterna.

11. Mentre OR restituisce TRUE se una delle due condizioni sono verificate

12. Gli operatori OR e IN vengono comunemente utilizzati nella condizione di ricerca (WHERE).
    OR restituisce TRUE se una delle due condizioni sono verificate, mentre IN restituisce TRUE quando il valore corrisponde a uno degli elementi specificati nell'elenco."

No, l'operatore logico BETWEEN restituisce TRUE solo per il range compreso tra gli estremi

CREAZIONE DATA BASE:

```
-- creazione del database
create database Prova_Finale_Database;
use Prova_Finale_Database;

-- creazione delle tabelle con vincoli della primary key e foreign key

-- drop table Product;
create table Product(
JoyID int primary key
, JoyName varchar(50)
, UnitePrice decimal (5,2));


create Table Category(
CategoryID int primary Key
,JoyID int
, Category varchar(50)
,constraint FK_Category_Product foreign key (JoyID)
references Product (JoyID));


-- drop table Region;
create table Region(
RegionID varchar(10) primary key
, Country varchar(50)
, Region varchar(50));

create table Address(
AddressLine varchar(100)
, Cap int
, Province char(5)
, RegionID varchar(10)
, constraint FK_Address_Region foreign key (RegionID)
references Region (RegionID));

-- drop table Sales;
create table  Sales(
TransactionID int primary key
, JoyID int
, DateTransaction date
, RegionID varchar(10)
, Quantity int
, SalesAmount decimal (10,2)
, constraint FK_Sales_Product foreign key (JoyID)
references Product (JoyID)
, constraint FK_Sales_Region foreign key (RegionID)
```

```
references Region (RegionID));

-- popolazione delle tabelle precedentemente create

insert into Product values
(1,'Velvet Matte','19.99')
,(2,'Glow Highlighter Palette','24.00')
,(3,'Ultra Volume Mascara','32.75')
,(4,'Perfect Match Foundation,'40');

insert into Region values
('201','North Italy','Italy')
,('202','Paris','France')
,('203','Berlin','Germany')
,('204','Madrid','Spain')


insert into Sales values
(501, 1, '01/04/25', '201', 15, 299.85),
(502, 2, '05/04/25, '202', 8, 276.00),
(503    3       10/04/25        203     12      288.00,
(504    4       12/04/25        204     5       199.50,

INSERT INTO Address VALUES
('Via Milano, 10'),
('Rue Lafayette,24'),
('Alexanderplatz, 3'),
('Calle de Alcalá,25'),

insert into category values
('101, 1, Lipstick')
,('102, 2, Highlighter')
,('103, 3,        Mascara')
,('104, 4, Foundation');

-----------------------------------------------------------------------------------------------------------------------------------------------------

1)Verificare che i campi definiti come PK siano univoci.
In altre parole, scrivi una query per determinare l'univocità dei valori di ciascuna PK
(una query per tabella implementata).
*/
USE CREAZIONE_DATABASE_E_TABELLE;

 SELECT JoyID
 , count(JoyID)
 FROM Product
 GROUP BY JoyID
 HAVING JoyID > 1;
```

```
SELECT TransactionID
, count(TransactionID)
FROM Sales
GROUP BY TransactionID
HAVING TransactionID > 1;

SELECT RegionID
, count(RegionID)
FROM Region
GROUP BY RegionID
HAVING RegionID > 1;

SELECT CategoryID
, count(CategoryID)
FROM Category
GROUP BY CategoryID
HAVING CategoryID > 1;

/*2)Esporre l'elenco delle transazioni indicando nel result set il codice documento,
la data, il nome del prodotto, la categoria del prodotto, il nome dello stato, il nome
della regione di venditae un campo booleano valorizzato in base alla condizione
che siano passati più di 180 giorni dalla data vendita o meno (>180 -> True, <= 180 -> False)*/

select
S.TransactionID
, S.DateTransaction
, P.JoyName
, C.Category
, R.Country
, R.Region
, CASE
      WHEN CURRENT_DATE - DateTransaction > 180 THEN TRUE
      ELSE FALSE
   END AS oltre_180_giorni
from Sales as S
inner join Product as P
on S.JoyID = P.JoyID
inner join Region as R
on S.RegionID = R.RegionID
inner join Category as C
on S.JoyID = C.JoyID;

/*3)Esporre l'elenco dei prodotti che hanno venduto, in totale,
una quantità maggiore della media delle vendite realizzate nell'ultimo anno censito. (2014)
(ogni valore della condizione deve risultare da una query e non deve essere inserito a mano).
Nel result set devono comparire solo il codice prodotto e il totale venduto.
*/
```

```
select JoyID
, Quantity
from sales
where Quantity > ( Select
        avg (Quantity)
                        from sales
        where Year(DateTransaction)=2014);
```

-- 4)Esporre l'elenco dei soli prodotti venduti e per ognuno di questi il fatturato totale per anno.

```
Select
JoyID
, extract(year from DateTransaction) as anno
-- , year(SalesAmount)
, sum(SalesAmount)
From sales
Group by JoyID, DateTransaction;
```

-- 5)Esporre il fatturato totale per stato per anno. Ordina il risultato per data e per fatturato decrescente.

```
select
S.JoyID
, extract(year from S.DateTransaction) as anno
, R.Country as Stato
, Sum(SalesAmount) as Tot_fatturato
from sales as S
Left join Region as R
on S.RegionID = R.RegionID
group by S.JoyID, R.Country, S.DateTransaction
order by S.DateTransaction, Tot_fatturato desc;
```

-- 6)Rispondere alla seguente domanda: qual è la categoria di articoli maggiormente richiesta dal mercato?

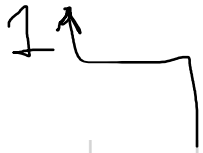
```
Select
Category
, sum(S.Quantity) as Tot_venduto
from Sales as S
Join Product as P
on S.JoyID = P.JoyID
join Category as C
on S.JoyID = C.JoyID
group by C.Category
order by Tot_venduto desc
limit 1;
```

```
-- 7)Rispondere alla seguente domanda: quali sono i prodotti invenduti? Proponi due approcci risolutivi differenti.
-- 1.
Select P.JoyID
,P.JoyName
From Product as P
left join Sales as S
on P.JoyID = S.JoyID
where S.TransactionId is null;
-- 2
Select P.JoyID
,Count(S.JoyID)
From sales as S
right join Product as P
on S.JoyID = P.JoyID
Group by P.JoyID
having Count(S.JoyID)<1;

/*
8)	Creare una vista sui prodotti in modo tale da esporre una “versione denormalizzata” delle informazioni utili (codice prodotto, nome prodotto, nome categoria)
*/

CREATE VIEW Versione_Denormalizzata as (
Select
P.JoyID as Codice
, P.JoyName as Nome
, C.Category as Categoria
From Product as P
join Category as C
on P.JoyID = C.JoyID);

-- 9)Creare una vista per le informazioni geografiche

CREATE VIEW Informazioni_Geografice as (
Select
R.regionID
, R.Region as Regione
, R.Country as Area
, A. addressLine as Via
, A.cap
, A.Province as Provincia
From Region as R
Inner Join Address as A
on R.RegionID = A.RegionID);
```

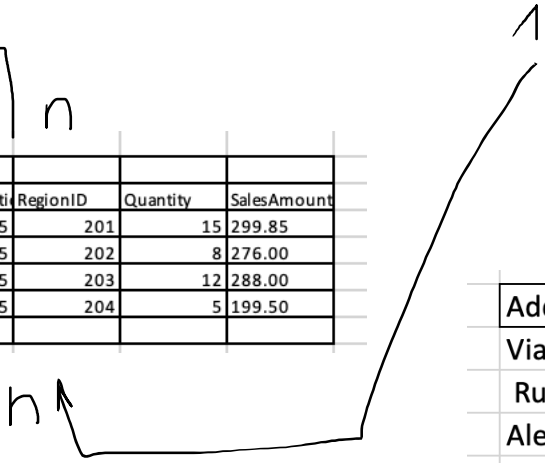
CONNESIONI – RELAZIONI FRA TABELLE:

Category		
CategoryID	JoyID	Category
101	1	Lipstick
102	2	Highlighter
103	3	Mascara
104	4	Foundation

Product		
JoyID	JoyName	UnitePrice
1	Velvet Matte L	19.99
2	Glow Highlig	34.50
3	Ultra Volume	24.00
4	Perfect Match	39.90

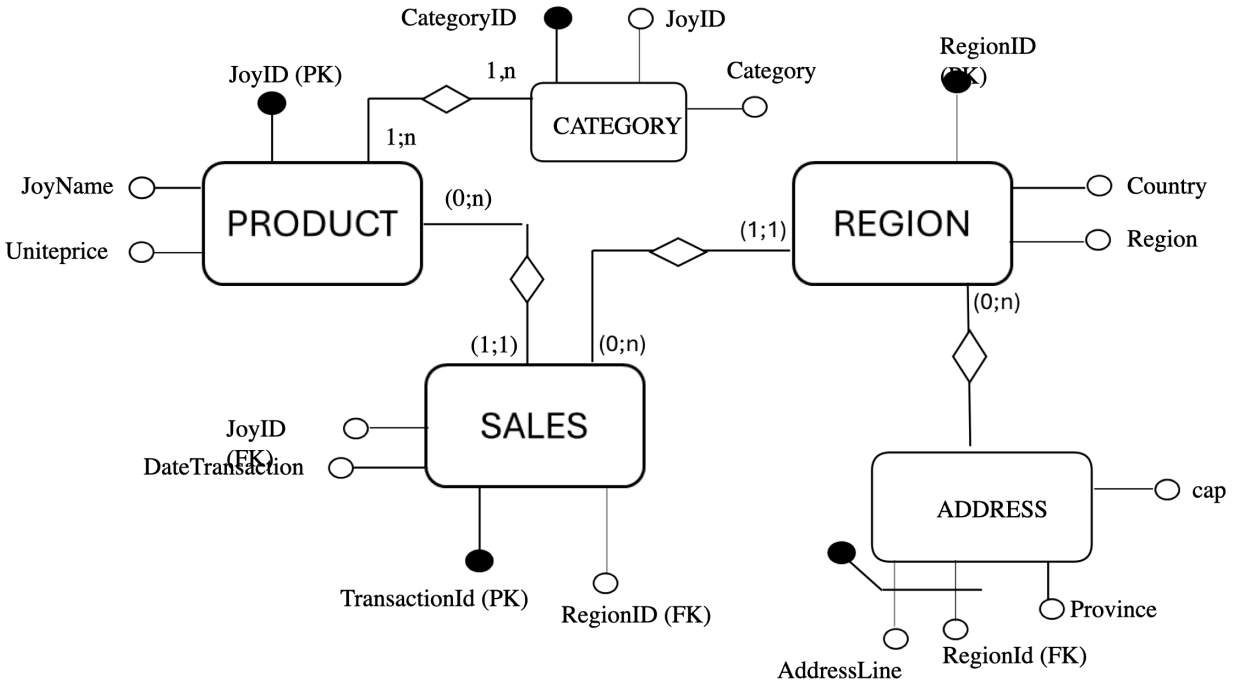


Region		
RegionID	Region	Country
201	North Italy	Italy
202	Paris	France
203	Berlin	Germany
204	Madrid	Spain



Sales					
TransactionID	JoyID	DateTransacti	RegionID	Quantity	SalesAmount
501	1	01/04/25	201	15	299.85
502	2	05/04/25	202	8	276.00
503	3	10/04/25	203	12	288.00
504	4	12/04/25	204	5	199.50

Address		
Via Milano	10	
Rue Lafayett	24	
Alexanderpla	3	
Calle de Alca	25	


CategoryID
JoyID
JoyID (PK)
1,n
Category
CATEGORY
1;n
RegionID
JoyName
PRODUCT
(0;n)
(1;1)
REGION
Country
Uniteprice
Region
(0;n)
(1;1)
(0;n)
SALES
JoyID
(FK)
DateTransaction
ADDRESS
cap
TransactionId (PK)
RegionID (FK)
Province
AddressLine
RegionId (FK)