**Questions**

1. Cosa si intende per database?

Database è un mero contenitore di dati, una raccolta di tabelle e relazioni (OLTP).

1. Cos’è un DBMS?

DBMS o database management system è sistema software che consente la creazione, la manipolazione e l'interrogazione efficiente di database.

1. Indica le principali clausole di uno statement SELECT in ordine di esecuzione logica. Descrivi per ciascuna delle clausole indicate la logica di funzionamento.

Statement Select sono un insieme di funzioni che consentono di ottenere un risultato (result set) e sono:

* FROM: consente di indicare la sorgente dati/tabella che vogliamo interrogare. Può essere arricchita da una join.
* WHERE: consente di applicare un filtro sulle righe della tabella fornita dalla FROM.
* GROUP BY: consente di raggruppare le righe della tabella filtrata per ogni combinazione univoca della group by list.
* HAVING: simile alla group by, consente di applicare un filtro sui raggruppamenti restituiti dalla group by
* SELECT: consente di selezione/indicare i campi desiderati nel result set.
* ORDER BY: consente di indicare un ordinamento (ascendete o discendente) del risultato della query.

1. Descrivi il concetto di group by con un esempio familiare. Utilizza l’approccio che ritieni più efficiente per trasmettere il concetto (suggerimento: disegna anche una sola tabella in excel o in word con poche colonne e pochi record e descrivi, basandoti sulla tabella stessa, un esempio di group by).

|  |  |  |
| --- | --- | --- |
| **PRODOTTO** | **DATA** | **QUANTITA’** |
| Prodotto\_A | 18/11/2023 | 17.00 |
| Prodotto\_B | 15/12/2022 | 51.50 |
| Prodotto\_B | 16/08/2023 | 23.20 |
| Prodotto\_A | 16/07/2022 | 26.80 |

Qual è la quantità totale per prodotto?

SELECT Prodotto, SUM(Quantità) AS Quantità\_totale

FROM Vendite

GROUP BY Prodotto

|  |  |
| --- | --- |
| **PRODOTTO** | **Quantità\_totale** |
| Prodotto\_A | 43.80 |
| Prodotto\_B | 74.70 |

1. Descrivi la differenza tra uno schema OLTP e uno schema OLAP.

OLPT, online transaction processing, è un database le cui tabelle (normalizzato) sono organizzate in maniera tale da ottimizzare le operazioni CRUD (create, read, update, delete) e garantire la minimizzazione dello spazio e l’integrità referenziale.

OLAP, online analytical processing, è un database relazionale generalmente denormalizzato che permette di ottimizzare l’analisi dei dati.

1. Dato un medesimo scenario di analisi, qual è la differenza in termini di risultato ottenibile tra una join e una subquery?

Join: istruzione che consente di combinare record di più tabelle in base alla corrispondenza dei valori di un campo in comune tra due tabelle

Subquery: è una query nidificata all’interno di un’altra query e ci permette di restituire un risultato che viene utilizzato per filtrare le righe della query principale.

1. Cosa si intende per DML e DDL?

DML: data manipulation language, include le istruzioni per interrogare e modificare i dati (select, insert, update, delete).

DDL: data definition language, include le funzioni che usiamo per gestire la struttura del database (create, alter, drop).

1. Quali istruzioni possono utilizzare per estrarre l’anno da un campo data? Proponi degli esempi.

Usiamo la funzione “YEAR”.

Esempio:

SELECT Max(DataInserimento)

FROM Ordini

WHERE YEAR(DataInserimento) = 2023

1. Qual è la differenza tra gli operatori logici AND e OR?

AND: restituisce TRUE quando entrambi le espressioni sono TRUE.

OR: restituisce TRUE quando una delle due condizioni è TRUE.

1. È possibile innestare una query nella clausola SELECT?

Si.

1. Qual è la differenza tra l’operatore logico OR e l’operatore logico IN?

OR: restituisce TRUE quando una delle due condizioni è TRUE.

IN: verifica se un valore è presente in un insieme di valori.

1. L’operatore logico BETWEEN include anche gli estremi del range specificato?

Si.

1. Che cos’è una SELF JOIN? Proponi un esempio.

È un tipo di join che ci permette di unire una tabella con se stessa.

**Case Study**

ToysGroup è un’azienda che distribuisce articoli (giocatoli) in diverse aree geografiche del mondo.

I prodotti sono classificati in categorie e i mercati di riferimento dell’azienda sono classificati in regioni di vendita.

In:

1. Le entità individuabili in questo scenario sono le seguenti:

* Product
* Region
* Sales

1. Le relazioni tra le entità possono essere descritte nel modo seguente:

* Product e Sales
* Un prodotto puo’ essere venduto tante volte (o nessuna) per cui è contenuto in una o più transazioni di vendita.
* Ciascuna transazione di vendita è riferita ad uno solo prodotto
* Region e Sales
* Possono esserci molte o nessuna transazione per ciascuna regione
* Ciascuna transazione di vendita è riferita ad una sola regione

1. Le entità Product e Region presentano delle gerarchie:

* L’entità prodotto contiene, oltre alle informazioni del singolo prodotto, anche la descrizione della categoria di appartenenza. L’entità prodotto contiene quindi una gerarchia: un prodotto puo’ appartenere ad una sola categoria mentre la stessa categoria puo’ essere associata a molti prodotti diversi.

*Esempio: gli articoli ‘Bikes-100’ e ‘Bikes-200’ appartengono alla categoria Bikes; gli articoli ‘Bike Glove M’ e ‘Bike Gloves L’ sono classificati come Clothing.*

* L’entità regione contiene una gerarchia: più stati sono classificati in una stessa regione di vendita e una stessa regione di vendita include molti stati.

*Esempio: gli stati ‘France’ e ‘Germany’ sono classificati nella region WestEurope; gli stati ‘Italy’ e ‘Greece’ sono classificati nel mercato SouthEurope*.

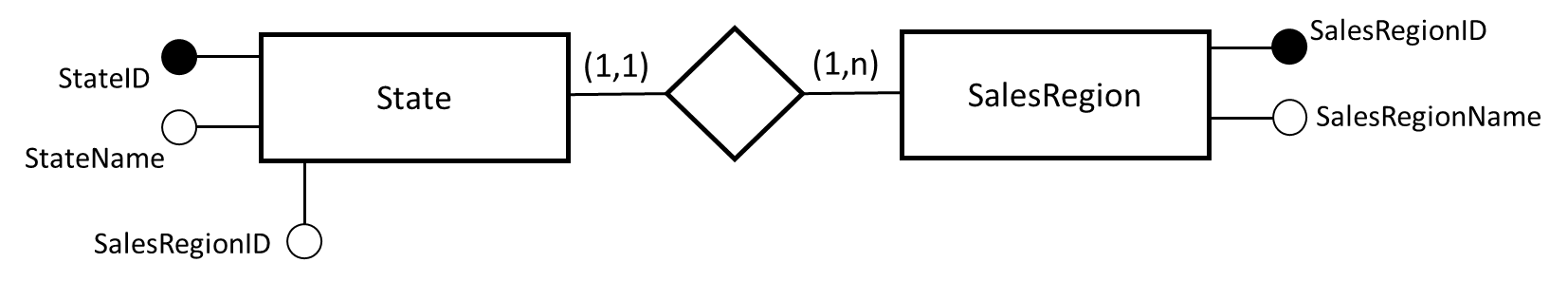
È necessario progettare e implementare fisicamente un database che modelli lo scenario garantendo l’**integrità referenziale** e la **minimizzazione della ridondanza dei dati.**

In altre parole, progetta opportunamente un numero di tabelle e di relazioni tra queste sufficiente a garantire la **consistenza del dato.**

**Task 1: Proponi una progettazione concettuale e logica della base dati**

La progettazione concettuale deve includere tutte le entità coinvolte e le relazioni tra queste. Per ciascuna entità indica l’attributo chiave e i principali attributi descrittivi (non è necessario indicare tutti gli attributi).

*Esempio di schema E/R*



*Lo schema proposto è puramente esemplificativo e non esaustivo o completo per la soluzione!*

La progettazione concettuale deve includere, per ciascuna tabella, tutte le colonne che poi verranno implementate fisicamente e deve esplicitare la cardinalità dei campi utilizzati per definire la relazione.

*Esempio di schema grafico delle tabelle e delle relazioni tra le stesse.*



n

1



*Il diagramma è puramente esemplificativo e non esaustivo o completo per la soluzione!*

**Task 2: Descrivi la struttura delle tabelle che reputi utili e sufficienti a modellare lo scenario proposto tramite la sintassi DDL. Implementa fisicamente le tabelle utilizzando il DBMS MySQL (o altro).**

CREATE TABLE table\_name (

column1 datatype option,

column2 datatype option,

column3 datatype,

....

);

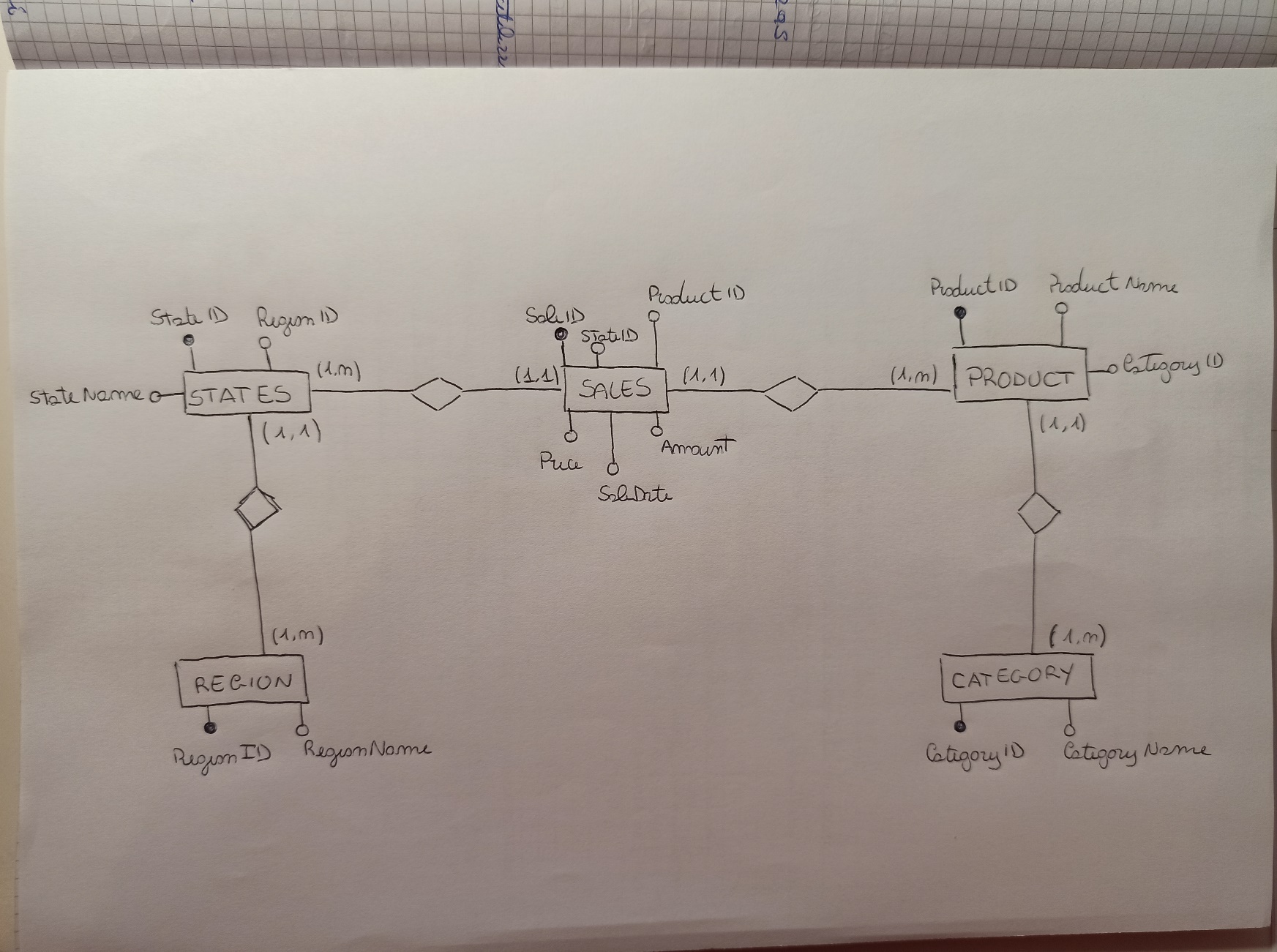
**Task 3: Popola le tabelle utilizzando dati a tua discrezione (sono sufficienti pochi record per tabella; riporta le query utilizzate)**

**Task 4: Dopo aver popolate le tabelle, scrivi delle query utili a:**

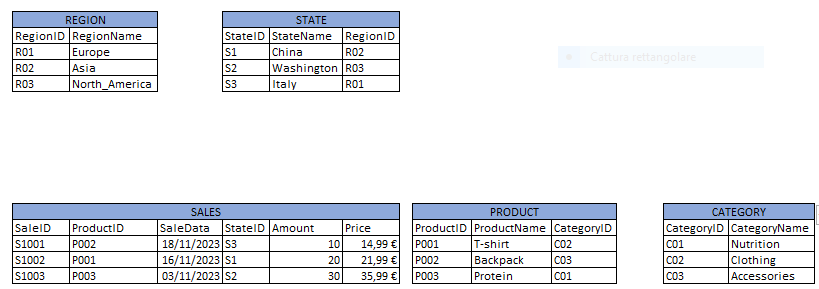
1. Verificare che i campi definiti come PK siano univoci. In altre parole, scrivi una query per determinare l’univocità dei valori di ciascuna PK (una query per tabella implementata).
2. Esporre l’elenco delle transazioni indicando nel result set il codice documento, la data, il nome del prodotto, la categoria del prodotto, il nome dello stato, il nome della regione di vendita e un campo booleano valorizzato in base alla condizione che siano passati più di 180 giorni dalla data vendita o meno (>180 -> True, <= 180 -> False)
3. Esporre l’elenco dei prodotti che hanno venduto, in totale, una quantità maggiore della media delle vendite realizzate nell’ultimo anno censito. (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.
4. Esporre l’elenco dei soli prodotti venduti e per ognuno di questi il fatturato totale per anno.
5. Esporre il fatturato totale per stato per anno. Ordina il risultato per data e per fatturato decrescente.
6. Rispondere alla seguente domanda: qual è la categoria di articoli maggiormente richiesta dal mercato?
7. Rispondere alla seguente domanda: quali sono i prodotti invenduti? Proponi due approcci risolutivi differenti.
8. Creare una vista sui prodotti in modo tale da esporre una “versione denormalizzata” delle informazioni utili (codice prodotto, nome prodotto, nome categoria)
9. Creare una vista per le informazioni geografiche

**TASK 1**

**PROGETTAZIONE CONCETTUALE**

****

**PROGETTAZIONE LOGICA**

****

**TASK 2**

*Creazione tabella “Category”:*

CREATE TABLE Category (

CategoryID VARCHAR(25),

CategotyName VARCHAR(255),

CONSTRAINT PK\_Category PRIMARY KEY (CategoryID)

);

*Creazione tabella “Product”:*

CREATE TABLE Product (

ProductID VARCHAR(25) PRIMARY KEY,

ProductName VARCHAR(255),

CategoryID VARCHAR(25),

CONSTRAINT FK\_Category\_CategoryID FOREIGN KEY (CategoryID)

REFERENCES Category (CategoryID)

);

*Creazione tabella “Region”:*

CREATE TABLE Region (

RegionID VARCHAR(25) PRIMARY KEY,

RegionName VARCHAR(255)

);

*Creazione tabella “State”:*

CREATE TABLE State (

StateID VARCHAR(25) PRIMARY KEY,

StateName VARCHAR(255),

RegionID VARCHAR(25),

CONSTRAINT FK\_Region\_RegionID FOREIGN KEY (RegionID)

REFERENCES Region (RegionID)

);

*Creazione tabella “SALES”:*

CREATE TABLE Sales (

SaleID VARCHAR(25) PRIMARY KEY,

ProductID VARCHAR(25),

SaleDate DATE,

StateID VARCHAR(25),

Amount INT,

Price DECIMAL(10 , 2 ),

CONSTRAINT FK\_Product\_ProductID FOREIGN KEY (ProductID)

REFERENCES Product (ProductID),

CONSTRAINT FK\_State\_StateID FOREIGN KEY (StateID)

REFERENCES State (StateID)

);

**TASK 3**

*TABELLA “CATEGORY”*

INSERT INTO Category (CategoryID, CategotyName)

VALUES

('C01', 'Nutrition'),

('C02', 'Clothing'),

('C03', 'Accessories');

*TABELLA “PRODUCT”*

INSERT INTO Product (ProductID, ProductName, CategoryID)

VALUES

('P001', 'T-shirt', 'C02'),

('P002', 'Backpack', 'C03'),

('P003', 'Protein', 'C01')

*TABELLA “REGION”*

INSERT INTO Region (RegionID, RegionName)

VALUES

('R01', 'Europe'),

('R02', 'Asia'),

('R03', 'North\_America');

*TABELLA “STATE”*

INSERT INTO State (StateID, StateName, RegionID)

VALUES

('S1', 'China', 'R02'),

('S2', 'Washington', 'R03'),

('S3', 'Italy', 'R01');

*TABELLA “SALES”*

INSERT INTO Sales (SaleID, ProductID, SaleDate, StateID, Amount, Price)

VALUES

('S1001', 'P002', '2023-11-18', 'S3', 10, 14.99),

('S1002', 'P001', '2023-11-16', 'S1', 20, 21.99),

('S1003', 'P003', '2023-11-03', 'S2', 30, 35.99);

**TASK 4**

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).

SELECT

CategoryID, COUNT(\*)

FROM

category

GROUP BY CategoryID

HAVING COUNT(\*) > 1;

SELECT

ProductID, COUNT(ProductID)

FROM

product

GROUP BY ProductID

HAVING COUNT(ProductID) > 1;

SELECT

RegionID, COUNT(RegionID)

FROM

region

GROUP BY RegionID

HAVING COUNT(RegionID) > 1;

SELECT

SaleID, COUNT(SaleID)

FROM

sales

GROUP BY SaleID

HAVING COUNT(SaleID) > 1;

1. 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 vendita e 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

SaleID,

SaleDate,

ProductName,

CategotyName,

StateName,

RegionName,

DATEDIFF(CURDATE(), S.SaleDate) > 180 AS Passati180Giorni

FROM

sales AS S

INNER JOIN

product AS P ON S.ProductID = P.ProductID

INNER JOIN

category AS C ON P.CategoryID = C.CategoryID

INNER JOIN

state AS St ON S.StateID = St.StateID

INNER JOIN

region AS R ON St.RegionID = R.RegionID;

1. Esporre l’elenco dei prodotti che hanno venduto, in totale, una quantità maggiore della media delle vendite realizzate nell’ultimo anno censito. (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

P.ProductID, SUM(Amount) AS TotaleVendite

FROM

sales AS S

JOIN

product AS P ON S.ProductID = P.ProductID

WHERE

SaleDate >= DATE\_SUB(CURDATE(), INTERVAL 1 YEAR)

GROUP BY P.ProductID

HAVING TotaleVendite >= (SELECT

AVG(Amount)

FROM

sales

WHERE

saledate >= DATE\_SUB(NOW(), INTERVAL 1 YEAR));

1. Esporre l’elenco dei soli prodotti venduti e per ognuno di questi il fatturato totale per anno.

SELECT

P.ProductID,

SUM(Price) AS FatturatoTotale,

YEAR(saledate) AS Anno

FROM

sales AS S

JOIN

product AS P ON S.ProductID = P.ProductID

GROUP BY ProductID , YEAR(SaleDate);

1. Esporre il fatturato totale per stato per anno. Ordina il risultato per data e per fatturato decrescente.

SELECT

St.StateName,

SUM(Price \* Amount) AS FatturatoTotale,

YEAR(saledate)

FROM

sales AS S

JOIN

state AS St ON S.StateID = St.StateID

GROUP BY StateName , YEAR(saledate)

ORDER BY YEAR(saledate) , FatturatoTotale DESC;

1. Rispondere alla seguente domanda: qual è la categoria di articoli maggiormente richiesta dal mercato.

SELECT

CategotyName, MAX(Amount) AS TotaleVendite

FROM

sales AS S

JOIN

product AS P ON S.ProductID = P.ProductID

JOIN

category AS C ON P.CategoryID = C.CategoryID

GROUP BY C.CategotyName

ORDER BY TotaleVendite DESC

LIMIT 1;

1. Rispondere alla seguente domanda: quali sono i prodotti invenduti? Proponi due approcci risolutivi differenti.

SELECT

ProductID, ProductName

FROM

product

WHERE

ProductID NOT IN (SELECT

productid

FROM

sales);

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

CREATE VIEW VistaProdotti AS

SELECT

ProductID, ProductName, C.CategoryID

FROM

product AS P

JOIN

category AS C ON P.CategoryID = C.CategoryID;

1. Creare una vista per le informazioni geografiche

CREATE VIEW GeographyView AS

SELECT

StateID, StateName, R.RegionName

FROM

state AS St

JOIN

region AS R ON St.RegionID = R.RegionID;