Domanda 1

- è un motore che gestisce la creazione e l'interrogazione dei database
- È ottimizzato per gestire l'interrogazione del database in maniera efficiente

Domanda 2

- decimal

Domanda 3

La funzione SELECT serve per recuperare dei dati memorizzati all'interno di un database.

Esempio:

```
SELECT *
FROM users
WHERE username='Elisa' AND password='Salve'
```

La funzione INSERT serve per inserire dati all'interno di una tabella.

Esempio:

```
INSERT INTO users (username, password)
 VALUES ('Elisa', 'Salve')
```

La funzione UPDATE permette di modificare e di aggiornare i record esistenti in una tabella.

Esempio:

```
UPDATE users
 SET username='eli' WHERE id=1
```

La funzione DELETE serve per cancellare dati in tabella.
Esempio:

```
DELETE FROM users
WHERE id=1
```

Domanda 4.

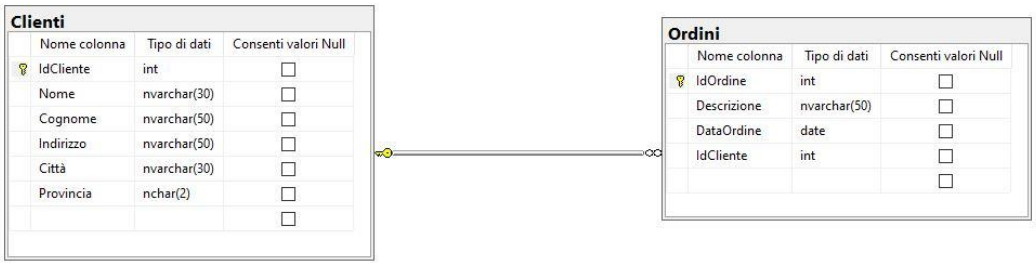
I vantaggi che si possono ottenere utilizzando una stored procedure piuttosto che una query SQL sono:

- riduzione del traffico dati tra server/client;

- miglioramento della sicurezza (assegnazioni ad hoc per utenti);
- Utilizzo per operazioni ripetitive senza dover riscrivere ogni volta il codice;
- Facilità nel modificare il codice (basta modificare la stored procedure);

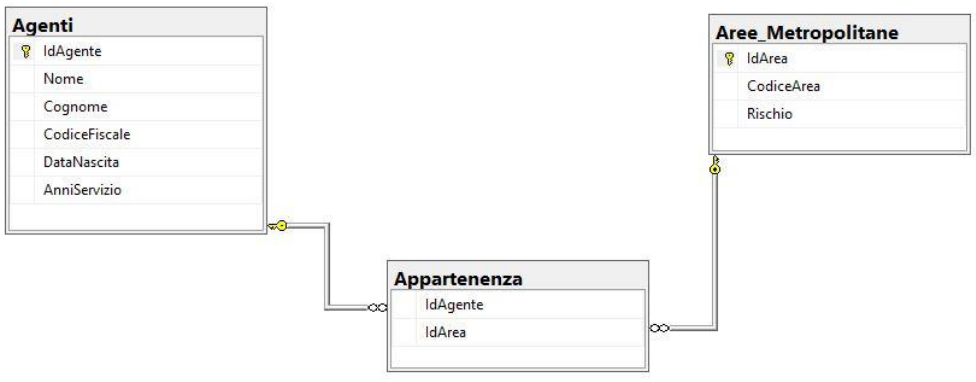
Domanda 5

La relazione 1: N si ha quando ad un'istanza dell'entità A corrispondono più istanze dell'entità B, mentre ad un'istanza dell'entità B corrisponde una sola istanza dell'entità A. Esempio:



Nell'esempio sopraindicato un cliente può effettuare più ordini ma un determinato ordine può essere effettuato solo da un cliente. L'IdCliente rappresenta la chiave primaria nella tabella Clienti, così come l'IdOrdine rappresenta la chiave primaria della tabella Ordini. Il campo chiave esterna IdCliente nella tabella Ordini è invece progettato per legare tra di loro i dati delle due tabelle.

Una relazione N:N si ha invece quando a un'istanza dell'entità A corrispondono più istanze dell'entità B e viceversa. Esempio:



Nell'esempio suindicato un agente può appartenere a più aree metropolitane e un'area metropolitana può essere sorvegliata a sua volta da più agenti. In genere i database di tipo relazionale non consentono di implementare una relazione "molti a

molti" diretta tra due tabelle. Per risolvere tale problematica è possibile utilizzare una tabella di appoggio chiamata anche tabella associativa che contenga al suo interno le chiavi primarie delle due tabelle che deve unire.

Domanda 6.

Una tabella in genere contiene uno o più attributi i cui valori identificano in modo univoco ogni record presente all'interno della tabella. Tali colonne prendono il nome di Primary Key (Chiave primaria) e garantiscono l'integrità di entità della tabella. Una primary key consente di correlare una tabella alle Foreign keys (Chiavi Esterne) di altre tabelle. La foreign key è una colonna o combinazione di colonne i cui valori corrispondono alla primary key di un'altra tabella. Esse vengono utilizzate per legare due tabelle tra di loro. Le foreign keys a differenza delle primary Keys non devono essere necessariamente univoche.

Domanda 7.

```
SELECT        Alunni.IdAlunno, Alunni.Nome, Alunni.Cognome, CONVERT(nchar(1),
dbo.Classi.Anno) + dbo.Classi.Sezione AS Classe
FROM            Alunni INNER JOIN Classi ON Alunni.IdClasse = Classi.IdClasse
```

Domanda 8.

Group by è una clausola di aggregazione che serve a specificare quali sono i campi su cui effettuare i raggruppamenti

Esempio:

SELECT Classe,

FROM STUDENTI

GROUP BY Classe

Attraverso questa istruzione è possibile vedere le classi che sono presenti all'interno della tabella Studenti

Domanda 9.

```
CREATE PROCEDURE ModificaUser
       @idUser int,
       @name nvarchar(50),
       @surname nvarchar(50),
       @birthday date
AS
BEGIN
```

```
	UPDATE users SET name=@name, surname=@surname, birthdate=@birthday WHERE
Id=@idUser


END

GO
```