Basi di Dati

Esercitazione 3: Interrogazioni in SQL

DB di riferimento per esempi

Consideriamo i seguenti schemi di una base di dati relazionale:

MODELLI(cod_modello, nome, versione, cod_fabbrica)

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)

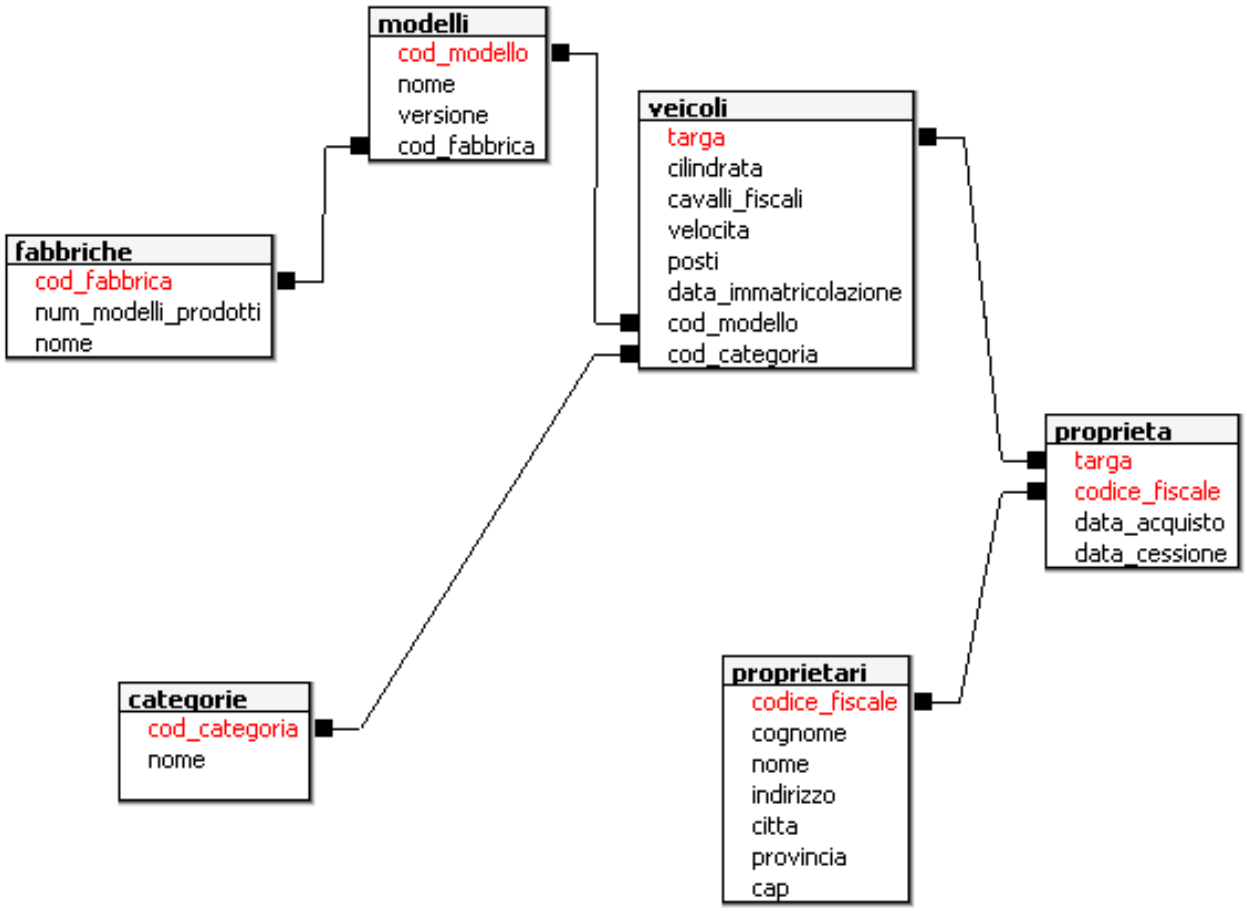
FABBRICHE(cod_fabbrica, nome, num_modelli_prodotti)

CATEGORIE(cod_categoria, nome)

PROPRIETARI(codice_fiscale, cognome, nome, indirizzo, citta, provincia, cap)

PROPRIETA(targa, codice_fiscale, data_acquisto, data_cessione)

DB di riferimento



Le chiavi primarie PK sono evidenziate in **rosso**.

Clausola GROUP BY

A volte può essere richiesto di calcolare operatori aggregati non per l'intera tabella, ma **raggruppando** le righe i cui valori coincidono su **un certo attributo**.

La clausola **GROUP BY** permette di specificare come dividere la tabella in sottoinsiemi.

Utilizzo di HAVING e WHERE con GROUP BY

Se la condizione coinvolge un attributo

- si usa la clausola **WHERE**

Se coinvolge un operatore aggregato

- si usa la clausola **HAVING**

Funzioni SUBSTRING e EXTRACT

- Per ottenere dalla stringa s la sottostringa lunga y caratteri partendo dal carattere di indice x, si utilizza:

SUBSTRING(s **FROM** x **FOR** y)

- Per ottenere rispettivamente **l'anno, il mese e il giorno** da una data, si utilizza:

EXTRACT(k FROM data**)** **≡** **date_part(x, data)**

Dove:
- **k** è parola chiave di SQL (**k = YEAR, MONTH** e **DAY)**
- data è un attributo di tipo Date
- **x=** 'year', 'month', 'day'

Subquery

Una "**subquery**" è una query inclusa in un‘ altra, ovvero un'interrogazione all'interno di altre interrogazioni

Il risultato di una subquery può essere:

- **un solo valore**
- **una colonna**
- **una tabella**

Il risultato di una subquery può essere usato nell’ esecuzione di query ad un livello più alto.

Regole per Subquery

➢ Le subquery oltre che essere usate all'interno della clausola **WHERE**, possono anche essere utilizzate nel calcolo di espressioni, dunque per **definire colonne**.

Esempio: Per ogni veicolo rappresentare la targa, la cilindrata e la differenza fra la cilindrata e la cilindrata minima

SELECT targa, cilindrata, cilindrata – (**SELECT MIN**(cilindrata)
FROM Veicoli) **AS** Differenza
FROM Veicoli;

➢ Con le subquery è possibile effettuare il confronto con tutti i valori di una colonna.

Per fare questo è bisogna utilizzare un operatore di confronto (=, <, >, >=, <=,...) seguito dalle parole chiave **ALL** ed **ANY**, seguite da una subquery.

Predicato EXISTS e NOT EXISTS

Tramite il predicato **EXISTS** è possibile effettuare il controllo sull'esistenza di righe che soddisfano specifiche condizioni.

In questo caso la subquery ritorna come risultato una tabella.

Si tratta di un **operatore logico** che assume il valore:
- **TRUE**: se la subquery dà un risultato non vuoto
- **FALSE**: se la subquery non dà un risultato non vuoto

Per verificare l'assenza di righe che rispondono a una determinata condizione è definita la forma negativa **NOT EXISTS**

JOIN

Il **NATURAL JOIN** esegue il join fra le tabelle specificate con la clausola FROM, di tutte le colonne che hanno lo stesso nome e stesso dominio.

L' **INNER JOIN** viene utilizzato quando non abbiamo nomi di colonna comuni fra le tabelle da sottoporre a join, oppure quando vogliamo esplicitamente dichiarare le condizioni di join. (la condizione non è necessariamente una condizione di uguaglianza)

Possiamo usere le forme:

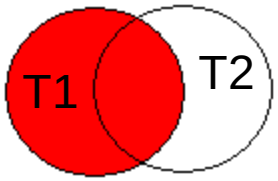
- T1 **JOIN** T2 **ON** CondizioneDiJoin
- T1 **JOIN** T2 **USING** (attr1,attr2,...)

Con **USING** viene specificato su quali colonne effettuare il JOIN

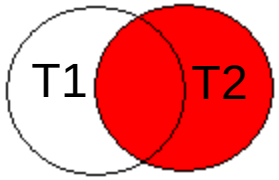
JOIN

L' **OUTER JOIN** estende il risultato di un semplice Join.
Ritorna tutte le righe che soddisfano la condizione di Join e quelle righe di una tabella per cui nessuna riga dell'altra tabella soddisfa la condizione di Join (estendendole con valori nulli).

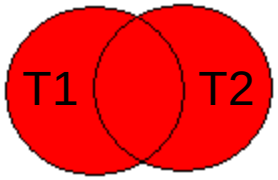
(sinistro): mantiene tutte le tuple del primo operando
T1 **LEFT [OUTER] JOIN** T2



(destro): mantiene tutte le tuple del secondo operando
T1 **RIGHT [OUTER] JOIN** T2



(completo): mantiene tutte le tuple di entrambi gli operandi
T1 **FULL [OUTER] JOIN** T2



Viste

Le Viste o **View** sono delle **tabelle virtuali** i cui dati sono aggregazioni dei dati contenuti nelle tabelle “fisiche” presenti nel database.

I dati sono contenuti nelle tabelle fisiche.
Le **View** forniscono una diversa visione dei dati presenti nelle tabelle fisiche.

La vista appare all’utente come una normale tabella, in cui può effettuare interrogazioni e, limitatamente ai suoi privilegi, anche modifiche dei dati.

Viste

I dati che vengono modificati su ciascuna delle tabelle fisiche utilizzate nella definizione della vista comporterà una modifica del contenuto della vista.

Equivalentemente, se i dati vengono modificati in una vista risulteranno modificati anche nella tabella che contiene fisicamente il dato.

- Viste Aggiornabili, soggette a restrizioni.

Viste

Gli utilizzi più comuni delle Viste sono per:

- Semplificare la rappresentazione dei dati
- Protezione dati
- Scomposizione query complesse

Viste

Sintassi:

CREATE [OR REPLACE] VIEW NomeVista [(ListaAttributi)]
AS
Select

Esempio 1: JOIN

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)
MODELLI(cod_modello, nome, versione, cod_fabbrica)

Trovare il nome del modello del veicolo di targa DF7281

```
SELECT nome
FROM Modelli m, Veicoli v
WHERE m.cod_modello = v.cod_modello AND v.targa = 'DF7281';
```

con Join esplicito:

```
SELECT nome
FROM Modelli NATURAL JOIN Veicoli
WHERE targa = 'DF7281';
```

Esempio 1: JOIN

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)
MODELLI(cod_modello, nome, versione, cod_fabbrica)

Trovare il nome del modello del veicolo di targa DF7281

con Join esplicito:

```
SELECT nome
FROM Modelli INNER JOIN Veicoli
USING (cod_modello)
WHERE targa = 'DF7281';
```

≡

```
SELECT nome
FROM Modelli m JOIN Veicoli v
ON m.cod_modello = v.cod_modello
WHERE targa = 'DF7281';
```

Esempio 2: JOIN

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)
CATEGORIE(cod_categoria, nome)

Trovare per ciascun veicolo la descrizione (nome) della relativa categoria utilizzando l'operatore INNER JOIN.

```
SELECT targa, v.cod_categoria, nome AS "Descrizione"
FROM Categorie c INNER JOIN Veicoli v
ON v.cod_categoria = c.cod_categoria;
```

Esempio 3: JOIN

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)
CATEGORIE(cod_categoria, nome)

Trovare per ciascun veicolo la descrizione (nome) della relativa categoria utilizzando l'operatore NATURAL JOIN.

```
SELECT targa, v.cod_categoria, nome AS "Descrizione"
FROM Categorie c NATURAL JOIN Veicoli v;
```

Esempio 4: JOIN

PROPRIETA(targa, codice_fiscale, data_acquisto, data_cessione)
PROPRIETARI(codice_fiscale, cognome, nome, indirizzo, citta, provincia, cap)

Visualizzare **tutti** i proprietari e i loro veicoli ordinati per cognome.

```
SELECT p.cognome, p.Nome, pr.targa
FROM Proprietari p LEFT JOIN Proprieta pr
ON p.codice_fiscale = pr.codice_fiscale
ORDER BY p.cognome;
```

cognome character(30)	nome character(30)	targa character(10)
Bianchi	Mario	
Luca	Paolo	
Neri	Luca	
Neri	Marco	
Rossi	Bianca	
Rossi	Paolo	BB18366J
Verdi	Giulia	AA123356X
Verdi	Giulia	BB18366J

Esempio 5:

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)
CATEGORIE(cod_categoria, nome)

Visualizzare per ogni categoria il numero di veicoli

```
SELECT c.cod_categoria, COUNT(*) AS "Numero veicoli"
FROM Veicoli v, Categorie c
WHERE v.cod_categoria = c.cod_categoria
GROUP BY c.cod_categoria;
```

Esempio 6:

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)
CATEGORIE(cod_categoria, nome)

Visualizzare per ogni categoria il numero di veicoli usando il JOIN esplicito

```
SELECT c.cod_categoria, COUNT(*) AS "Numero veicoli"
FROM Veicoli v INNER JOIN Categorie c
ON v.cod_categoria = c.cod_categoria
GROUP BY c.cod_categoria;
```

Esempio 7:

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)
CATEGORIE(cod_categoria, nome)

Visualizzare per ogni categoria il numero di veicoli:

```
SELECT Categorie.cod_categoria, COUNT(*) AS "Numero veicoli"
FROM Veicoli NATURAL INNER JOIN Categorie
GROUP BY Categorie.cod_categoria
```

Esempio 8:

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)

Per ogni categoria estrarre la velocita media dei veicoli.

```
SELECT cod_categoria, AVG(velocita) AS "Media Velocita"
FROM Veicoli
GROUP BY cod_categoria;
```

Esempio 9:

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)

Per ogni categoria estrarre la velocita media dei veicoli con 5 posti.

```
SELECT cod_categoria, AVG(velocita) AS "Media Velocita"
FROM Veicoli
WHERE posti = 5
GROUP BY cod_categoria;
```

Esempio 10:

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)

Per ogni categoria la cui media della velocità è > 100, estrarre la velocita media.

```
SELECT cod_categoria, AVG(velocita) AS "Media Velocita"
FROM Veicoli
GROUP BY cod_categoria
HAVING AVG(velocita)>100;
```

Esempio 11:

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)

Per ogni categoria con almeno due veicoli, estrarre la velocita media.

```
SELECT cod_categoria, AVG(velocita) AS "Media Velocita"
FROM Veicoli
GROUP BY cod_categoria
HAVING COUNT(*)>1;
```

Esercizio 12: INNER JOIN

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)
MODELLI(cod_modello, nome, versione, cod_fabbrica)

Visualizzare i modelli presenti e per ogni modello il numero di veicoli

```
SELECT m.cod_modello,  COUNT(*) AS "Numero veicoli"
FROM Veicoli v, Modelli m
WHERE v.cod_modello = m.cod_modello
GROUP BY m.cod_modello;
```

JOIN esplicito:

```
SELECT m.cod_modello, COUNT(*) AS "Numero veicoli"
FROM Veicoli v [INNER] JOIN Modelli m
ON v.cod_modello = m.cod_modello
GROUP BY m.cod_modello;
```

Esercizio 12: INNER JOIN

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)
MODELLI(cod_modello, nome, versione, cod_fabbrica)

Visualizzare il nome dei modelli presenti e per ogni modello il numero di veicoli

```
SELECT m.cod_modello, m.nome, COUNT(*) AS "Numero veicoli"
FROM Veicoli v, Modelli m
WHERE v.cod_modello = m.cod_modello
GROUP BY m.cod_modello, m.nome;
```

JOIN esplicito:

```
SELECT m.cod_modello, m.nome, COUNT(*) AS "Numero veicoli"
FROM Veicoli v [INNER] JOIN Modelli m
ON v.cod_modello = m.cod_modello
GROUP BY m.cod_modello, m.nome;
```

Esercizio 13:

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)
MODELLI(cod_modello, nome, versione, cod_fabbrica)
PROPRIETA(targa, codice_fiscale, data_acquisto, data_cessione)

Fornire l'elenco dei veicoli ceduti nell'anno 2003 specificando la targa, ed il nome del modello.

```
SELECT v.targa, m.nome "Nome Modello"
FROM modelli m, veicoli v
WHERE v.cod_modello = m.cod_modello AND
            v.targa IN
                    (SELECT targa
                     FROM proprieta
                     WHERE EXTRACT (YEAR FROM data_cessione) = 2003);
```

Esercizio 13:

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)
MODELLI(cod_modello, nome, versione, cod_fabbrica)
PROPRIETA(targa, codice_fiscale, data_acquisto, data_cessione)

Fornire l'elenco dei veicoli ceduti nell'anno 2003 specificando la targa, ed il nome del modello (utilizzando il **JOIN Esplicito**).

```
SELECT v.targa, m.nome "Nome Modello"
FROM modelli m [INNER] JOIN veicoli v
ON v.cod_modello = m.cod_modello
WHERE v.targa IN
        (SELECT targa
         FROM proprieta
         WHERE EXTRACT (YEAR FROM data_cessione) = 2003)
```

Esercizio 13:

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)
MODELLI(cod_modello, nome, versione, cod_fabbrica)
PROPRIETA(targa, codice_fiscale, data_acquisto, data_cessione)

Fornire l'elenco dei veicoli ceduti nell'anno 2003 specificando la targa, ed il nome del modello (utilizzando **NATURAL JOIN**).

```
SELECT v.targa, m.nome "Nome Modello"
FROM modelli m NATURAL JOIN veicoli v
WHERE v.targa IN
        (SELECT targa
         FROM proprieta
         WHERE EXTRACT (YEAR FROM data_cessione) = 2003)
```

Esempio 14: SUBSTRING(s FROM x FOR y)

FABBRICHE(cod_fabbrica, nome, num_modelli_prodotti)

Fornire (in ordine alfabetico per nome) le fabbriche il cui nome contenga tra le prime 10 lettere sia una 'A', sia una 'I' (non distinguendo tra maiuscole e minuscole).

```
SELECT nome
FROM fabbriche
WHERE SUBSTRING(nome FROM 1 FOR 10) ILIKE '%A%' AND
      SUBSTRING(nome FROM 1 FOR 10) ILIKE '%I%'
ORDER BY nome;
```

Esempio 15: EXTRACT(k FROM d)

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)

Visualizzare le targhe dei veicoli immatricolati a ottobre 2001

```
SELECT targa
FROM Veicoli
WHERE EXTRACT (MONTH FROM data_immatricolazione) = 10  AND
          EXTRACT (YEAR FROM data_immatricolazione) = 2001;
```

Visualizzare per tutti i veicoli la targa e l'anno di immatricolazione

```
SELECT targa, EXTRACT(YEAR FROM data_immatricolazione)
                    AS "Anno Immatricolazione"
FROM Veicoli;
```

Esempio 16:

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)

Trovare tutti i veicoli di cilindrata massima.

```
SELECT *
FROM Veicoli
WHERE Cilindrata = (SELECT MAX(cilindrata)
                    FROM Veicoli)
```

Esempio 17:

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)

Trovare tutti i veicoli di cilindrata superiore alla cilindrata media.

```
SELECT *
FROM Veicoli
WHERE Cilindrata > (SELECT AVG(Cilindrata)
                    FROM Veicoli)
```

Esempio 19:

MODELLI(cod_modello, nome, versione, cod_fabbrica)
VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)
FABBRICHE(cod_fabbrica, nome, num_modelli_prodotti)

Trovare targa e velocità dei veicoli che appartengono a Modelli prodotti nella Fabbrica FIAT (usando le subquery)

```
SELECT targa, velocita
FROM Veicoli
WHERE cod_modello IN
                    (SELECT cod_modello
                     FROM Modelli
                     WHERE cod_fabbrica =
                              (SELECT cod_fabbrica
                              FROM Fabbriche
                              WHERE nome='FIAT'))
```

Esempio 20:

MODELLI(cod_modello, nome, versione, cod_fabbrica)
VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)
FABBRICHE(cod_fabbrica, nome, num_modelli_prodotti)

Trovare targa e velocità dei veicoli che appartengono a Modelli prodotti nella Fabbrica FIAT (usando il JOIN implicito)

```
SELECT targa, velocita
FROM Veicoli v, Modelli m, Fabbriche f
WHERE v.cod_modello = m.cod_modello  AND
        m.cod_fabbrica = f.cod_fabbrica AND
        f.nome = 'FIAT';
```

Esempio 20:

MODELLI(cod_modello, nome, versione, cod_fabbrica)
VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)
FABBRICHE(cod_fabbrica, nome, num_modelli_prodotti)

Trovare targa e velocità dei veicoli che appartengono a Modelli prodotti nella Fabbrica FIAT (usando il JOIN esplicito)

```
SELECT targa, velocita
FROM Veicoli v INNER JOIN Modelli m USING(cod_modello)
INNER JOIN Fabbriche f USING(cod_fabbrica)
WHERE  f.nome = 'FIAT';
```

oppure

```
SELECT targa, velocita
FROM Veicoli v INNER JOIN Modelli m ON v.cod_modello = m.cod_modello
INNER JOIN Fabbriche f ON m.cod_fabbrica = f.cod_fabbrica
WHERE  f.nome = 'FIAT';
```

Esempio 21:

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)
CATEGORIE(cod_categoria, nome)

Trovare tutti i veicoli della categoria “AUTOVEICOLO”

Mediante JOIN

```
SELECT v.*
FROM Categorie c, Veicoli v
WHERE c.cod_categoria =v.cod_categoria AND c.nome = ‘AUTOVEICOLO’
```

Mediante Subquery

```
SELECT * FROM Veicoli
WHERE cod_categoria = (SELECT cod_categoria
                       FROM Categorie
                       WHERE nome = ‘AUTOVEICOLO’)
```

Esempio 22:

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)

Selezionare tutti i veicoli con cilindrata **inferiore ad almeno una** delle cilindrate dei veicoli con modello 003 (quindi inferiore alla più alta di queste cilindrate).

```
SELECT *
FROM Veicoli
WHERE cilindrata <= ANY (SELECT cilindrata
                         FROM Veicoli
                         WHERE cod_modello='003')
```

```
SELECT *
FROM Veicoli
WHERE cilindrata<= (SELECT MAX(cilindrata)
                    FROM Veicoli
                    WHERE cod_modello='003' )
```

Esempio 23:

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)

Selezionare tutti i veicoli con cilindrata **inferiore a tutte** le cilindrate dei veicoli con modello 002 (quindi inferiore alla più bassa di queste cilindrate)

```
SELECT *
FROM Veicoli
WHERE cilindrata < ALL (SELECT cilindrata
                        FROM Veicoli
                        WHERE cod_modello='002')
```

```
SELECT *
FROM Veicoli
WHERE cilindrata < (SELECT MIN(cilindrata)
                    FROM Veicoli
                    WHERE cod_modello='002')
```

Esempio 24:

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)
MODELLI(cod_modello, nome, versione, cod_fabbrica)

Trovare l' intersezione dei modelli con cilindrata inferiore a 1400 e quelli con codice fabbrica uguale a 001.

```
SELECT cod_modello
FROM Veicoli
WHERE cilindrata < 1400 AND
        cod_modello = ANY [IN] (SELECT cod_modello
                                FROM Modelli
                                WHERE cod_fabbrica = '001')
```

NOTA: **=ANY** è equivalente a **IN**

Esempio 25:

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)
MODELLI(cod_modello, nome, versione, cod_fabbrica)

Trovare i modelli con cilindrata inferiore a 1400 che non sono prodotti dalla fabbrica con codice 001

```
SELECT cod_modello
FROM Veicoli
WHERE cilindrata < 1400
AND cod_modello <> ALL [NOT IN] (SELECT cod_modello
                                 FROM Modelli
                                 WHERE cod_fabbrica='001')
```

NOTA: **<>ALL** è equivalente a **NOT IN**

Esempio 26:

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)
CATEGORIE(cod_categoria, nome)

Trovare i nomi di tutte le categorie per cui è presente almeno un veicolo

```
SELECT nome
FROM Categorie
WHERE EXISTS (SELECT *
              FROM Veicoli
              WHERE categorie.cod_categoria = Veicoli.cod_categoria )
```

Esempio 27:

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)
CATEGORIE(cod_categoria, nome)

Trovare tutte le categorie per cui non è presente nessun veicolo

```
SELECT nome
FROM Categorie
WHERE NOT EXISTS (SELECT *
                  FROM Veicoli
                  WHERE Categorie. cod_categoria = Veicoli. cod_categoria)
```

Equivalenze: EXISTS,=ANY,IN

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)
CATEGORIE(cod_categoria, nome)

```
SELECT nome
FROM Categorie
WHERE EXISTS (SELECT *
                    FROM Veicoli
                    WHERE categorie. cod_categoria = Veicoli.cod_categoria)

SELECT nome
FROM Categorie
WHERE cod_categoria =ANY (SELECT cod_categoria
                                  FROM Veicoli)

SELECT nome
FROM Categorie
WHERE cod_categoria IN (SELECT cod_categoria
                                FROM Veicoli)
```

Equivalenze: NOT EXISTS,<>ALL,NOT IN

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)
CATEGORIE(cod_categoria, nome)

```
SELECT nome
FROM Categorie
WHERE NOT EXISTS (SELECT *
                  FROM Veicoli
                  WHERE Categorie.cod_categoria=Veicoli. cod_categoria)
```

```
SELECT nome
FROM Categorie
WHERE cod_categoria <>ALL (SELECT cod_categoria
                           FROM Veicoli)
```

```
SELECT nome
FROM Categorie
WHERE cod_categoria NOT IN (SELECT cod_categoria
                            FROM Veicoli)
```

Esercizio 28: View

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)

Creare una vista che contiene la targa e la cilindrata dei veicoli con cilindrata <1500

```
CREATE [OR REPLACE] VIEW PiccolaCilindrata (Targa,cilindrata) AS
        SELECT targa, cilindrata
        FROM Veicoli
        WHERE cilindrata < 1500;
```

Esercizio 29: View

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)

Creare una vista che contiene la targa e la cilindrata dei motocicli (cod_categoria = '02') con cilindrata <150

```
CREATE [OR REPLACE] VIEW PiccolaCilindrata (Targa,cilindrata) AS
        SELECT targa, cilindrata
        FROM Veicoli
        WHERE cod_categoria = '02' AND cilindrata < 150;
```

Esercizio 30: View

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)
CATEGORIE(cod_categoria, nome)

Creare una vista che contiene la targa, il codice del modello e il nome della categoria dei veicoli.

```
CREATE [OR REPLACE] VIEW A1 AS
        SELECT targa, cod_modello, nome
        FROM Veicoli, Categorie
        WHERE Categorie.cod_categoria = Veicoli.cod_categoria;
```

NOTA:
Mancando la specifica dei nomi delle colonne della vista, vengono acquisiti i nomi delle colonne della tabella madre.

Esercizio 31: View

MODELLI(cod_modello, nome, versione, cod_fabbrica)

Creare una vista che per ciascuna fabbrica riporti il numero totale delle versioni dei modelli prodotti.

```
CREATE [OR REPLACE] VIEW A2 (cod_fabbrica, numero_versioni) AS
        SELECT cod_fabbrica, SUM(versione)
        FROM Modelli
        GROUP BY cod_fabbrica;
```

INSERT

Sintassi:

INSERT INTO nome tabella [(ListaAttributi)]
VALUES (ListaDiValori) | SELECT

INSERT INTO

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)
FABBRICHE(cod_fabbrica, nome, num_modelli_prodotti)

INSERT INTO fabbriche(cod_fabbrica, nome, num_modelli_prodotti)
VALUES ('006', 'DUCATI', 7);

INSERT INTO fabbriche(cod_fabbrica, nome)
VALUES ('007', 'APRILIA');

INSERT INTO veicoli(targa, cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)
VALUES ('CB263GR', 1200, 115, 160, 5, '2001-12-27', '001', '01');

INSERT INTO mediante SELECT

PROPRIETARI(codice_fiscale, cognome, nome, indirizzo, citta, provincia, cap)

```
INSERT INTO ElencoProprietariPI(codice_fiscale, nome, cognome)
        SELECT codice_fiscale, nome, cognome
        FROM Proprietari
        WHERE provincia = 'PI';
```

Attenzione lo schema della tabella ElencoProprietariPI deve essere creato prima:

```
CREATE TABLE ElencoProprietariPI (
        codice_fiscale CHAR(16),
        cognome text,
        nome text,
        PRIMARY KEY (codice_fiscale));
```

DELETE

Cancellazione di righe da tabelle

Sintassi:

DELETE FROM nome_tabella [**WHERE** Condizione]

DELETE

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)

Cancellare il veicolo con targa 'CB263GR'

DELETE FROM Veicoli **WHERE** targa = 'CB263GR';

Per cancellare tutto il contenuto di una tabella:

DELETE FROM Veicoli;

DELETE

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)

Eliminare tutti i veicoli non presenti nella tabella Proprieta

```
DELETE FROM Veicoli
WHERE targa NOT IN
                    (SELECT targa
                    FROM Proprieta)
```

UPDATE

Consente di aggiornare alcuni o tutti i dati.

Sintassi:

UPDATE Tabella
SET Attributo = Espressione
[**WHERE** Condizione]

UPDATE

VEICOLI(targa , cilindrata, cavalli_fiscali, velocita, posti, data_immatricolazione, cod_modello, cod_categoria)

Della tabella veicoli aggiornare

UPDATE veicoli
SET posti = 1
WHERE cod_categoria = ‘03’;