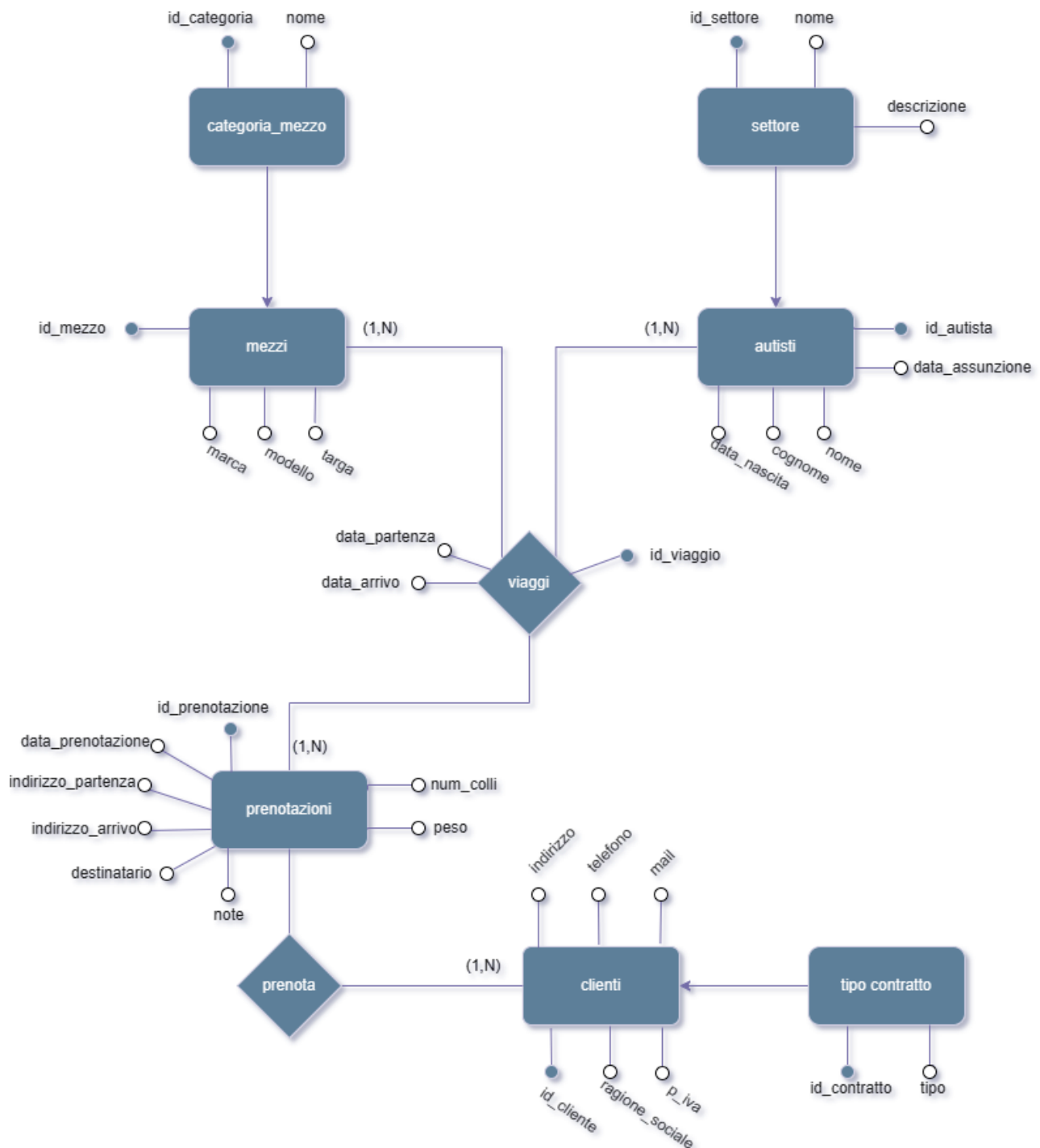


# Database centro Logistica

Il database che ho progettato è quello di un centro logistico con l'obiettivo di analizzare i dati riguardanti le prenotazioni effettuate dai clienti e i relativi viaggi ma anche di capire quali sono i tipi di mezzo e i settori di trasporto che vengono maggiormente utilizzati.

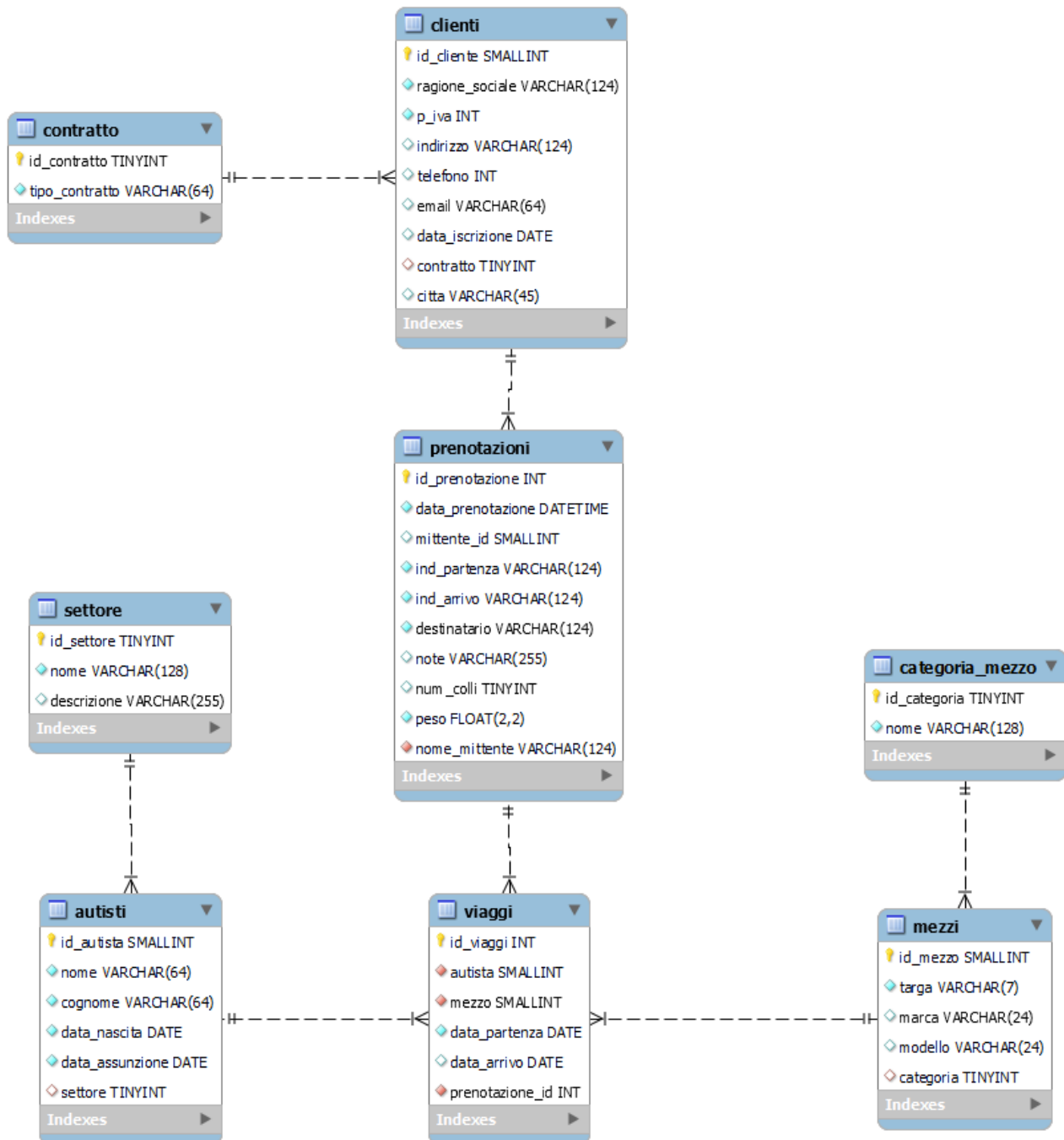
Di seguito il **modello concettuale** utilizzato:



Partendo dal modello concettuale ho poi ricavato il modello logico in forma tabellare su excel, per poi creare il db tramite Mysql command line.

Una volta creato il db l'ho caricato su Mysql workbench per poter lavorare sui dati più velocemente.

Infine, tramite la funzione "reverse engineering", ho ricavato il **modello logico** che segue:



Tornando alla creazione del database, ecco di seguito i passaggi effettuati:

Per prima cosa ho creato il db con il comando

```
CREATE database logistica;
```

per poi andare a creare le relative tabelle:

```
MySQL 8.0 Command Line Cli x + v
owners.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+-----+
| Database |
+-----+
| discografia |
| information_schema |
| logistica |
| m3d7 |
| mysql |
| performance_schema |
| sakila |
| sys |
| videogamestore |
| world |
+-----+
10 rows in set (0.01 sec)

mysql> use logistica;
Database changed
mysql> create table settore (
  -> id_settore tinyint primary key,
  -> nome varchar(128) not null,
  -> descrizione varchar(255) default "Nessuna descrizione") engine=innodb;
Query OK, 0 rows affected (0.05 sec)

mysql>
```

```
mysql> explain settore;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| id_settore | tinyint | NO | PRI | NULL | auto_increment |
| nome | varchar(128) | NO | | NULL | |
| descrizione | varchar(255) | YES | | Nessuna descrizione | |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql> create table categoria_mezzo(
  -> id_categoria tinyint auto_increment primary key,
  -> nome varchar(128) not null)
  -> ;
Query OK, 0 rows affected (0.02 sec)

mysql> create table contratto(
  -> id_contratto tinyint auto_increment primary key,
  -> tipo_contratto varchar(64) not null)
  -> ;
Query OK, 0 rows affected (0.02 sec)
```

```
mysql> create table mezzi(
-> id_mezzo smallint auto_increment primary key,
-> targa varchar(7) not null,
-> marca varchar(24) default "Nessuna descrizione",
-> modello varchar(24) default "Nessuna descrizione",
-> categoria tinyint,
-> foreign key(categoria) references categoria_mezzo(id_categoria) on update cascade on delete set null);
Query OK, 0 rows affected (0.04 sec)
```

```
mysql> explain mezzi;
```

Field	Type	Null	Key	Default	Extra
id_mezzo	smallint	NO	PRI	NULL	auto_increment
targa	varchar(7)	NO		NULL	
marca	varchar(24)	YES		Nessuna descrizione	
modello	varchar(24)	YES		Nessuna descrizione	
categoria	tinyint	YES	MUL	NULL	

```
5 rows in set (0.00 sec)
```

```
mysql> create table clienti(
-> id_cliente smallint auto_increment primary key,
-> ragione_sociale varchar(124) not null,
-> p_iva tinyint not null,
-> indirizzo varchar(124) default "Nessuna descrizione",
-> telefono tinyint default 0,
-> email varchar(64),
-> data_iscrizione date,
-> contratto tinyint,
-> foreign key(contratto) references contratto(id_contratto) on update cascade on delete set null);
Query OK, 0 rows affected (0.04 sec)
```

```
mysql> explain clienti;
```

Field	Type	Null	Key	Default	Extra
id_cliente	smallint	NO	PRI	NULL	auto_increment
ragione_sociale	varchar(124)	NO		NULL	
p_iva	tinyint	NO		NULL	
indirizzo	varchar(124)	YES		Nessuna descrizione	
telefono	tinyint	YES		0	
email	varchar(64)	YES		NULL	
data_iscrizione	date	YES		NULL	
contratto	tinyint	YES	MUL	NULL	

Una volta create tutte le tabelle con i relativi attributi e chiavi esterne ho cominciato ad inserire i dati:

```
mysql> insert into settore(nome,descrizione)
-> values("trasporto straordinario","trasporto sopra le 3,5 tonnellate");
Query OK, 1 row affected (0.01 sec)

mysql> insert into settore(nome,descrizione)
-> values("trasporto speciale","trasporto materiale liquido e/o rischio infiammabilità");
Query OK, 1 row affected (0.01 sec)
```

```
mysql> select * from settore;
```

id_settore	nome	descrizione
1	trasporto locale	trasporto sotto le 3,5 tonnellate
2	trasporto straordinario	trasporto sopra le 3,5 tonnellate
3	trasporto speciale	trasporto materiale liquido e/o rischio infiammabilità

```
3 rows in set (0.00 sec)
```

```
mysql> INSERT INTO autisti(nome,cognome,data_nascita,data_assunzione,settore)
-> VALUES("Giuseppe","Garibaldi","1985-06-14","2000-06-01",3);
Query OK, 1 row affected (0.01 sec)
```

```
mysql> SELECT * FROM autisti;
```

id_autista	nome	cognome	data_nascita	data_assunzione	settore
1	Fabrizio	Feliziani	1994-02-12	2010-01-01	1
2	Giuseppe	Garibaldi	1985-06-14	2000-06-01	3

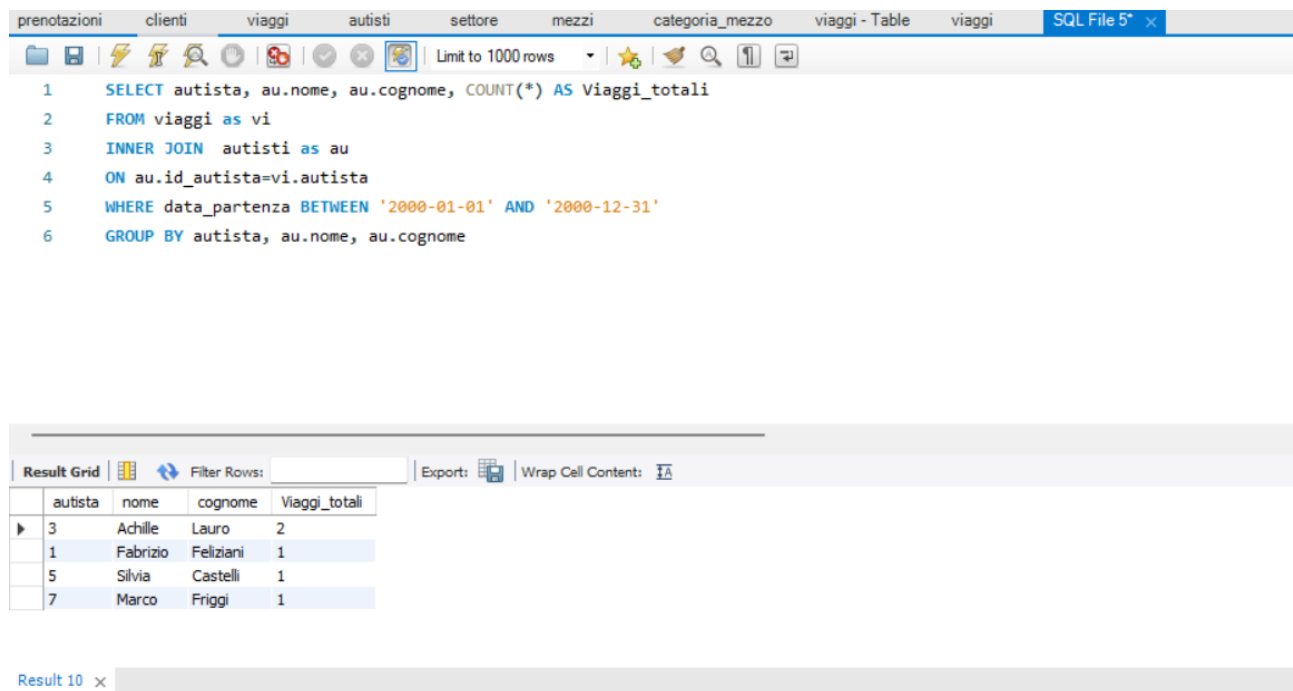
```
2 rows in set (0.00 sec)
```

Quelli sopra elencati sono alcuni esempi di inserimento effettuati. In secondo luogo, per far scorrere meglio il lavoro, ho importato su Mysql workbench i file .csv in cui avevo già creato dati di esempio. Una volta caricati tutti i dati ho cominciato con le interrogazioni.

---

### -- QUERY 1: Viaggi totali per ogni autista per l'anno 2000

```
SELECT autista, au.nome, au.cognome, COUNT(*) AS Viaggi_totali
FROM viaggi AS vi
INNER JOIN autisti AS au
ON au.id_autista=vi.autista
WHERE data_partenza BETWEEN '2000-01-01' AND '2000-12-31'
GROUP BY autista, au.nome, au.cognome;
```



The screenshot displays the MySQL Workbench interface. The top toolbar includes icons for file operations, execution, and navigation. The SQL editor window, titled 'SQL File 5\*', contains the following query:

```
1 SELECT autista, au.nome, au.cognome, COUNT(*) AS Viaggi_totali
2 FROM viaggi as vi
3 INNER JOIN autisti as au
4 ON au.id_autista=vi.autista
5 WHERE data_partenza BETWEEN '2000-01-01' AND '2000-12-31'
6 GROUP BY autista, au.nome, au.cognome
```

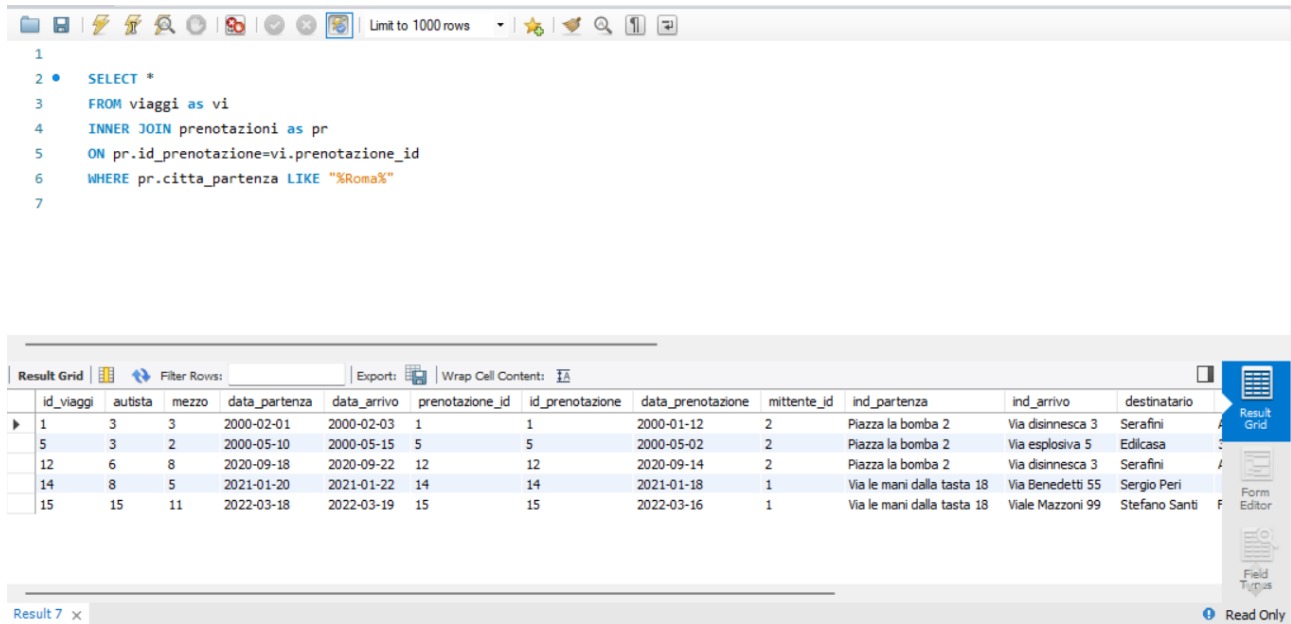
Below the editor, the 'Result Grid' tab is active, showing the query results in a table with 4 columns: autista, nome, cognome, and Viaggi\_totali. The results are as follows:

	autista	nome	cognome	Viaggi_totali
▶	3	Achille	Lauro	2
	1	Fabrizio	Feliziani	1
	5	Silvia	Castelli	1
	7	Marco	Friggi	1

At the bottom of the interface, a status bar indicates 'Result 10'.

## -- QUERY 2: Elenco dei trasporti partiti da una specifica città

```
SELECT *  
FROM viaggi AS vi  
INNER JOIN prenotazioni AS pr  
ON pr.id_prenotazione=vi.prenotazione_id  
WHERE pr.citta_partenza LIKE "%Roma%";
```



The screenshot shows a database query tool interface. At the top, there's a toolbar with icons for file operations, search, and execution. Below the toolbar, the SQL query is entered in a text area. The query is as follows:

```
1  
2 • SELECT *  
3 FROM viaggi as vi  
4 INNER JOIN prenotazioni as pr  
5 ON pr.id_prenotazione=vi.prenotazione_id  
6 WHERE pr.citta_partenza LIKE "%Roma%"  
7
```

Below the query editor, the results are displayed in a table grid. The grid has 13 columns: id\_viaggi, autista, mezzo, data\_partenza, data\_arrivo, prenotazione\_id, id\_prenotazione, data\_prenotazione, mittente\_id, ind\_partenza, ind\_arrivo, destinatario, and a final column with a partial label 'A'. The results show 5 rows of data.

id_viaggi	autista	mezzo	data_partenza	data_arrivo	prenotazione_id	id_prenotazione	data_prenotazione	mittente_id	ind_partenza	ind_arrivo	destinatario	A
1	3	3	2000-02-01	2000-02-03	1	1	2000-01-12	2	Piazza la bomba 2	Via disinnesca 3	Serafini	A
5	3	2	2000-05-10	2000-05-15	5	5	2000-05-02	2	Piazza la bomba 2	Via esplosiva 5	Edिकास	S
12	6	8	2020-09-18	2020-09-22	12	12	2020-09-14	2	Piazza la bomba 2	Via disinnesca 3	Serafini	A
14	8	5	2021-01-20	2021-01-22	14	14	2021-01-18	1	Via le mani dalla tasta 18	Via Benedetti 55	Sergio Peri	A
15	15	11	2022-03-18	2022-03-19	15	15	2022-03-16	1	Via le mani dalla tasta 18	Viale Mazzoni 99	Stefano Santi	F

At the bottom of the interface, there's a tab labeled 'Result 7' and a 'Read Only' status indicator.

**-- QUERY 3: Elenco autisti che hanno fatto più di 2 Viaggi**

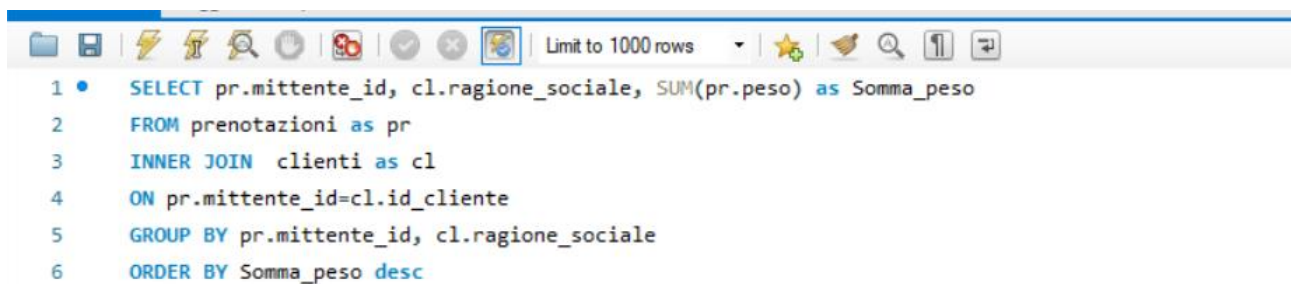
```
SELECT *  
FROM (SELECT COUNT(*) AS totale_viaggi, au.nome, au.cognome  
      FROM viaggi AS vi  
      INNER JOIN autisti AS au  
      ON au.id_autista=vi.autista  
      GROUP BY au.nome, au.cognome) AS tot  
WHERE totale_viaggi >=2;
```

```
1 • SELECT *  
2 FROM(  
3   SELECT COUNT(*) as totale_viaggi, au.nome, au.cognome  
4   FROM viaggi as vi  
5   INNER JOIN autisti as au  
6   ON au.id_autista=vi.autista  
7   GROUP BY au.nome, au.cognome) as tot  
8   WHERE totale_viaggi >=2;
```

Result Grid			
Filter Rows:		Export:	Wrap Cell Content:
	totale_viaggi	nome	cognome
▶	2	Fabrizio	Feliziani
	2	Achille	Lauro
	2	Andrea	Russo
	2	Alessandro	Alessandri

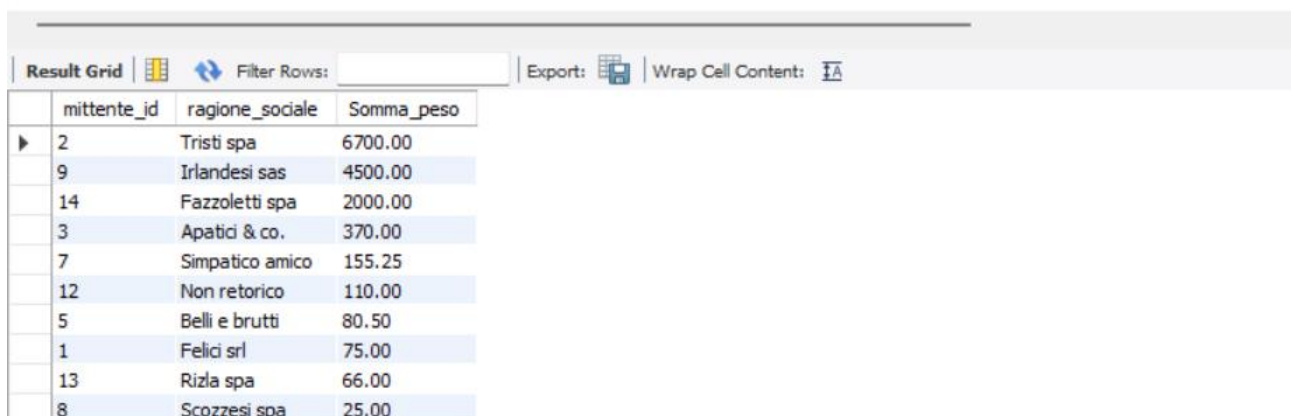
-- QUERY 4: Elenco dei clienti raggruppati per la somma totale del peso dei colli da loro spediti in ordine decrescente

```
SELECT pr.mittente_id, cl.ragione_sociale, SUM(pr.peso) AS Somma_peso  
FROM prenotazioni AS pr  
INNER JOIN clienti AS cl  
ON pr.mittente_id=cl.id_cliente  
GROUP BY pr.mittente_id, cl.ragione_sociale  
ORDER BY Somma_peso DESC;
```



The screenshot shows a SQL query editor with a toolbar at the top. The query is as follows:

```
1 • SELECT pr.mittente_id, cl.ragione_sociale, SUM(pr.peso) as Somma_peso  
2 FROM prenotazioni as pr  
3 INNER JOIN clienti as cl  
4 ON pr.mittente_id=cl.id_cliente  
5 GROUP BY pr.mittente_id, cl.ragione_sociale  
6 ORDER BY Somma_peso desc
```



The screenshot shows a database result grid with the following data:

	mittente_id	ragione_sociale	Somma_peso
▶	2	Tristi spa	6700.00
	9	Irlandesi sas	4500.00
	14	Fazzoletti spa	2000.00
	3	Apatici & co.	370.00
	7	Simpatico amico	155.25
	12	Non retorico	110.00
	5	Belli e brutti	80.50
	1	Felici srl	75.00
	13	Rizla spa	66.00
	8	Scozzesi spa	25.00



**-- QUERY 5: Il numero di autisti per ogni settore di trasporto**

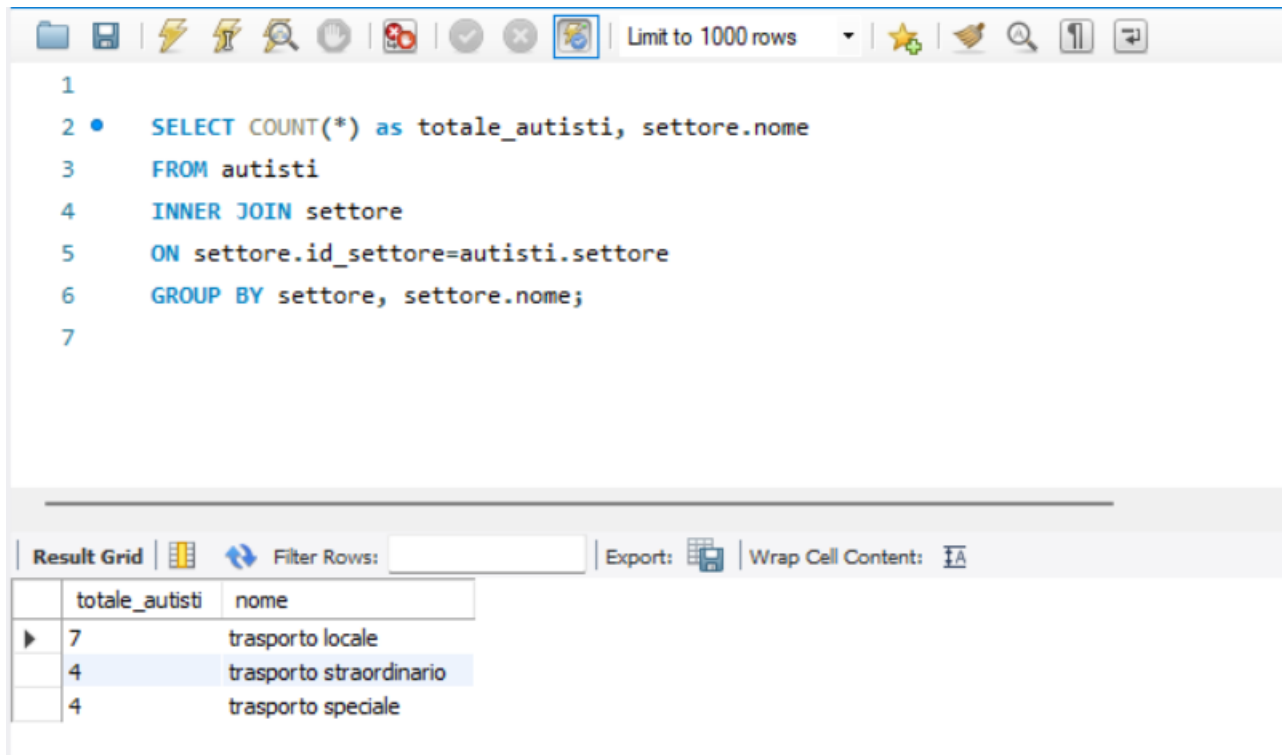
```
SELECT COUNT(*) AS totale_autisti, settore.nome
```

```
FROM autisti
```

```
INNER JOIN settore
```

```
ON settore.id_settore=autisti.settore
```

```
GROUP BY settore, settore.nome;
```



The screenshot shows a SQL query editor interface. The top toolbar includes icons for file operations, execution, and a 'Limit to 1000 rows' dropdown. The query editor contains the following SQL code:

```
1
2 • SELECT COUNT(*) as totale_autisti, settore.nome
3 FROM autisti
4 INNER JOIN settore
5 ON settore.id_settore=autisti.settore
6 GROUP BY settore, settore.nome;
7
```

Below the query editor is a results panel with a toolbar containing 'Result Grid', 'Filter Rows', 'Export', and 'Wrap Cell Content'. The results are displayed in a table:

	totale_autisti	nome
▶	7	trasporto locale
	4	trasporto straordinario
	4	trasporto speciale

**-- QUERY 6: Elenco dei mezzi categoria "Autobotte" partiti nel 2000**

```
SELECT mezzi.id_mezzo, mezzi.targa, mezzi.marca, mezzi.modello, mezzi.categoria,categoria_mezzo.nome,  
viaggi.data_partenza
```

```
FROM mezzi
```

```
INNER JOIN categoria_mezzo
```

```
ON id_categoria=mezzi.categoria
```

```
INNER JOIN viaggi
```

```
ON viaggi.mezzo=mezzi.id_mezzo
```

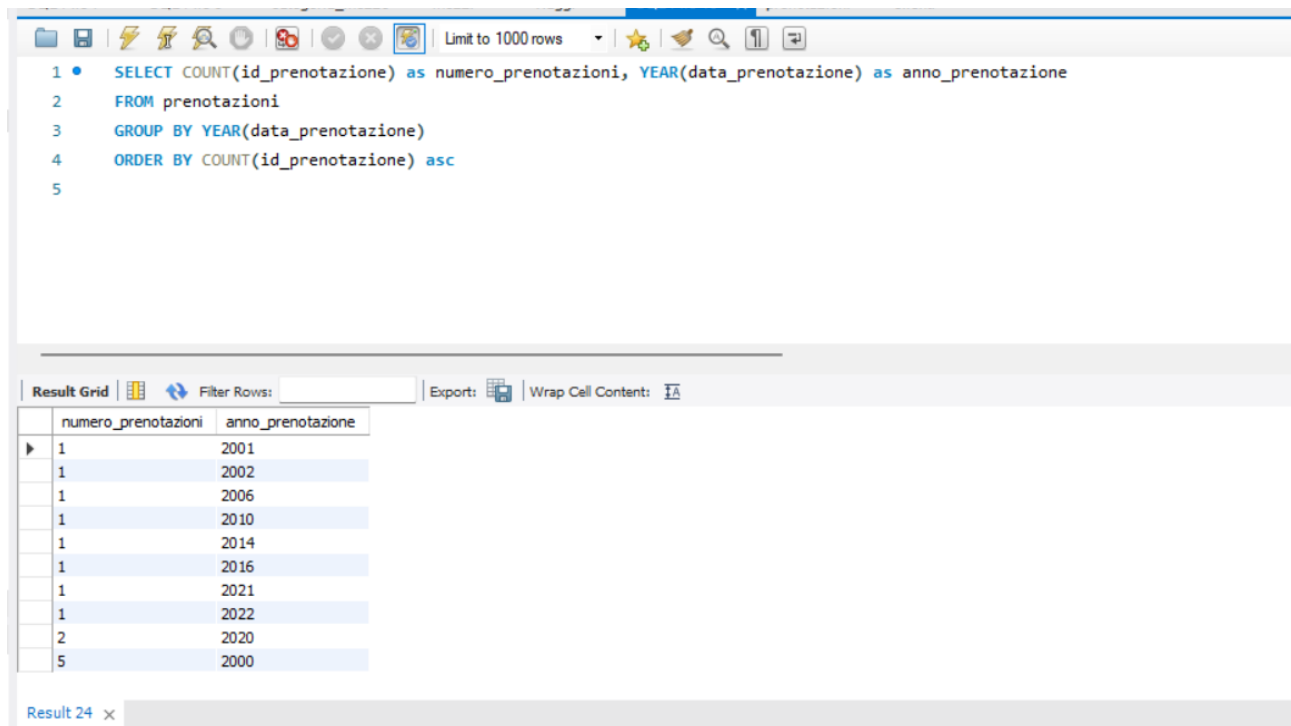
```
WHERE YEAR(data_partenza)="2000" AND categoria_mezzo.nome LIKE "%Autobotte%";
```

```
1  
2 • SELECT mezzi.id_mezzo, mezzi.targa, mezzi.marca, mezzi.modello, mezzi.categoria,categoria_mezzo.nome, viaggi.data_partenza  
3 FROM mezzi  
4 INNER JOIN categoria_mezzo  
5 ON id_categoria=mezzi.categoria  
6 INNER JOIN viaggi  
7 ON viaggi.mezzo=mezzi.id_mezzo  
8 WHERE YEAR(data_partenza)="2000" AND categoria_mezzo.nome LIKE "%Autobotte%"  
9
```

Result Grid							
Filter Rows:							
Export: Wrap Cell Content:							
	id_mezzo	targa	marca	modello	categoria	nome	data_partenza
▶	3	CC789DD	Mercedes	660	3	Autobotte	2000-02-01

**-- QUERY 7: Numero delle prenotazioni per ogni anno**

```
SELECT COUNT(id_prenotazione) as numero_prenotazioni, YEAR(data_prenotazione) as anno_prenotazione  
FROM prenotazioni  
GROUP BY YEAR(data_prenotazione)  
ORDER BY COUNT(id_prenotazione) asc;
```



The screenshot shows a SQL query editor with a toolbar at the top. The query is as follows:

```
1 • SELECT COUNT(id_prenotazione) as numero_prenotazioni, YEAR(data_prenotazione) as anno_prenotazione  
2 FROM prenotazioni  
3 GROUP BY YEAR(data_prenotazione)  
4 ORDER BY COUNT(id_prenotazione) asc  
5
```

Below the query editor, there is a 'Result Grid' section. It includes a 'Filter Rows' input field, an 'Export' button, and a 'Wrap Cell Content' checkbox. The results are displayed in a table with two columns: 'numero\_prenotazioni' and 'anno\_prenotazione'.

	numero_prenotazioni	anno_prenotazione
1	1	2001
1	1	2002
1	1	2006
1	1	2010
1	1	2014
1	1	2016
1	1	2021
1	1	2022
2	2	2020
5	5	2000

At the bottom left, there is a tab labeled 'Result 24' with a close button (X).

**-- QUERY 8: Clienti che hanno spedito più di 4 colli**

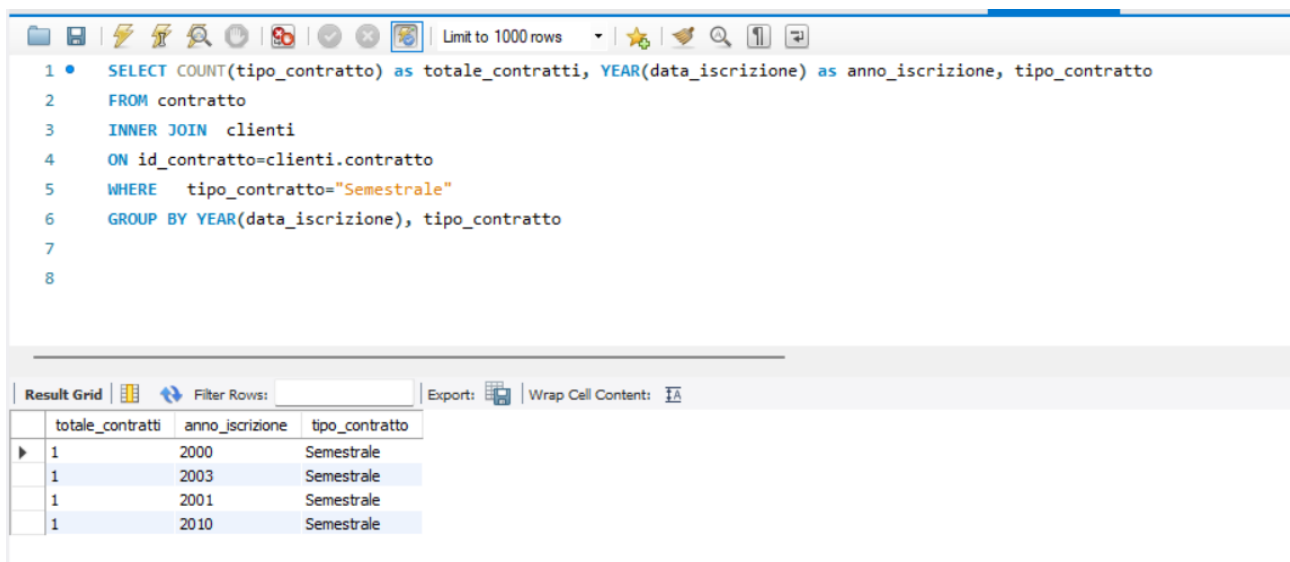
```
SELECT ragione_sociale, SUM(num_colli) as somma_colli  
FROM prenotazioni  
INNER JOIN clienti  
ON clienti.id_cliente=mittente_id  
GROUP BY ragione_sociale  
HAVING somma_colli >=4;
```

```
1 • SELECT ragione_sociale, SUM(num_colli) as somma_colli  
2 FROM prenotazioni  
3 INNER JOIN clienti  
4 ON clienti.id_cliente=mittente_id  
5 GROUP BY ragione_sociale  
6 HAVING somma_colli >=4  
7
```

Result Grid		Filter Rows:	Export:	Wrap Cell Content:
	ragione_sociale	somma_colli		
►	Tristi spa	5		
	Apatici & co.	4		
	Simpatico amico	5		
	Irlandesi sas	4		
	Belli e brutti	6		
	Non retorico	12		
	Felici srl	5		

**-- QUERY 9: Quanti contratti di tipo "Semestrale" sono stati fatti ogni anno**

```
SELECT COUNT(tipo_contratto) as totale_contratti, YEAR(data_iscrizione) as anno_iscrizione, tipo_contratto
FROM contratto
INNER JOIN clienti
ON id_contratto=clienti.contratto
WHERE tipo_contratto="Semestrale"
GROUP BY YEAR(data_iscrizione), tipo_contratto;
```



The screenshot shows a database query editor interface. The top toolbar includes icons for file operations, execution, and settings, along with a 'Limit to 1000 rows' dropdown. The SQL query is displayed in a text area, and the results are shown in a table below. The table has three columns: 'totale\_contratti', 'anno\_iscrizione', and 'tipo\_contratto'. The results show four rows, all with 'Semestrale' as the contract type and a count of 1 for each year (2000, 2003, 2001, 2010).

	totale_contratti	anno_iscrizione	tipo_contratto
1	1	2000	Semestrale
1	1	2003	Semestrale
1	1	2001	Semestrale
1	1	2010	Semestrale

**-- QUERY 10: Tutti i viaggi partiti in data odierna (12/03/2023) riportando nome e cognome autista, targa del mezzo, cliente mittente, città di partenza e arrivo**

```
SELECT id_viaggi, concat(au.nome," ",au.cognome) as nome_autista, targa, ragione_sociale, citta_partenza,
citta_arrivo, data_partenza

FROM viaggi

LEFT JOIN autisti as au

ON viaggi.autista=au.id_autista

LEFT JOIN mezzi as me

ON viaggi.mezzo=me.id_mezzo

LEFT JOIN (SELECT ragione_sociale, id_prenotazione

FROM clienti as cl

INNER JOIN prenotazioni as pr

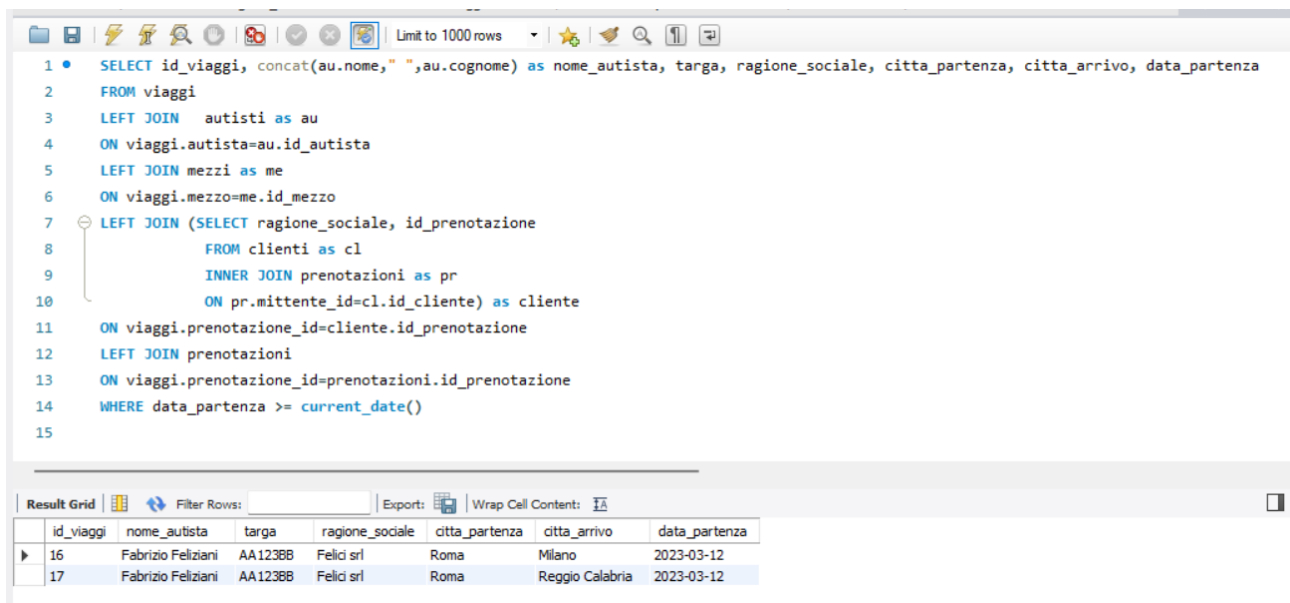
ON pr.mittente_id=cl.id_cliente) as cliente

ON viaggi.prenotazione_id=cliente.id_prenotazione

LEFT JOIN prenotazioni

ON viaggi.prenotazione_id=prenotazioni.id_prenotazione

WHERE data_partenza >= current_date();
```



The screenshot shows a SQL query editor with a toolbar at the top. The query is displayed in a text area, and below it is a 'Result Grid' showing the results of the query. The query is the same as the one in the previous block. The result grid has 7 columns: id\_viaggi, nome\_autista, targa, ragione\_sociale, citta\_partenza, citta\_arrivo, and data\_partenza. It contains two rows of data.

	id_viaggi	nome_autista	targa	ragione_sociale	citta_partenza	citta_arrivo	data_partenza
▶	16	Fabrizio Feliziani	AA123BB	Felici srl	Roma	Milano	2023-03-12
	17	Fabrizio Feliziani	AA123BB	Felici srl	Roma	Reggio Calabria	2023-03-12

Nel presente documento ho illustrato solo alcune delle analisi che potrebbero essere effettuate lavorando su un database del genere. Le analisi che ho effettuato sono, ovviamente, limitate ai pochi dati di cui disponevo. Per portare a termine questo progetto ho utilizzato alcune delle funzioni apprese durante il corso.

In allegato, nella cartella, ho inserito tutta la documentazione di cui mi sono servito: i grafici dei modelli, i file excel, il file dump del database creato e, inoltre, un file in cui ho elencato tutte le interrogazioni effettuate, così da poterle utilizzare direttamente sul db fornito.

Fabrizio Feliziani