Basi di Dati

Oracle SQLPLUS - Esercitazione n. 3

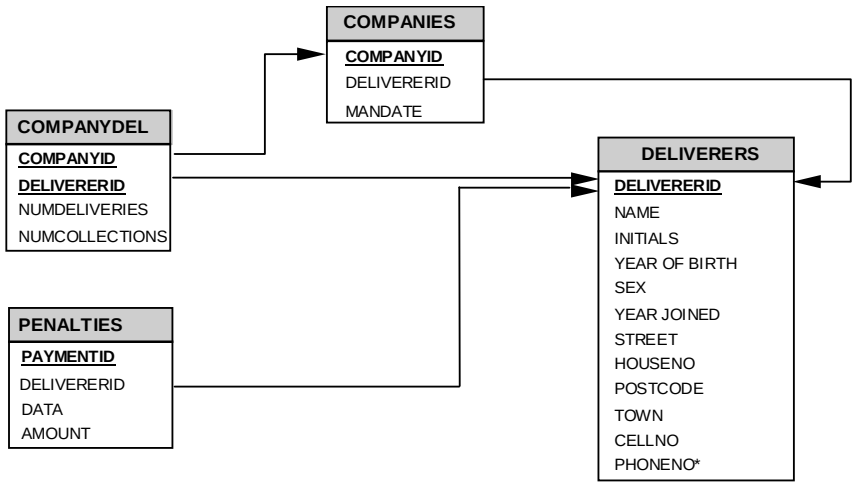
Questa esercitazione è composta da due parti. La parte I contiene alcune interrogazioni da risolvere usando il linguaggio SQL su una base di dati già esistente. La parte II ha come obiettivo quello di creare, dato lo schema logico di una base di dati, appositi script di creazione e popolamento della base di dati e di scrivere ed eseguire alcuni comandi di aggiornamento e cancellazione utilizzando il linguaggio SQL.

PARTE I

1. Descrizione del Database *Delivery*

Il database *Delivery* raccoglie informazioni relative alle attività svolte da una ditta di fattorini che svolge consegne e ritiri di merci per diverse aziende.
La tabella DELIVERERS contiene i dati anagrafici dei fattorini. In particolare, per ogni fattorino sono disponibili il codice identificativo (DELIVERERID), il nome, le inziali, l'anno di nascita, il sesso, l'anno di inizio attività, la via, il numero civico, la città, il codice postale del luogo di residenza, il numero di cellulare e l'interno dell'ufficio in cui è dislocato.
Nella tabella COMPANYDEL sono riportati i dati relativi alle consegne e ai ritiri fatti da ogni fattorino presso le varie aziende (identificate dal codice COMPANYID). Per ogni coppia fattorino-azienda per cui è avvenuta almeno una consegna o un ritiro sono noti il numero di consegne (NUMDELIVERIES) e di ritiri (NUMCOLLECTIONS) effettuati.
La tabella PENALTIES raccoglie le multe ricevute dai fattorini. Per ogni multa vengono memorizzati il codice della multa (PENALTYID), il codice del fattorino, la data e l'importo da pagare.
Nella tabella COMPANIES per ogni azienda è noto il codice identificativo dell'azienda (COMPANYID) e il codice identificativo del referente ad essa assegnato (un fattorino) e il numero di mandati per cui il referente attuale ha ricoperto tale carica (MANDATE).

La struttura della base di dati è riportata nella figura seguente e i dati contenuti nelle tabelle sono riportati nella Sezione 2.



2. Contenuto delle Tabelle del Database *Delivery*

3. La chiave primaria è sottolineata. I campi che possono assumere il valore nullo sono contrassegnati dal simbolo *.

Tabella **DELIVERERS**

DELIVERERID	NAME	INITIALS	YEAR_OF_BIRTH	SEX	YEAR JOINED	STREET	HOUSENO	POSTCODE	TOWN	CELLNO	PHONENO*
2	Everett	R	1948	M	1975	Stoney Road	43	3575NH	Stratford	070-237893	2411
6	Parmenter	R	1964	M	1977	Haseltine Lane	80	1234KK	Stratford	070-476537	8467
7	Wise	GWS	1963	M	1981	Edgecombe Way	39	9758VB	Stratford	070-347689	***NULL***
8	Newcastle	B	1962	F	1980	Station Road	4	6584WO	Inglewood	070-476573	2983
27	Collins	DD	1964	F	1983	Long Drive	804	8457DK	Eltham	079-234857	2513
28	Collins	C	1963	F	1983	Old main Road	10	1294QK	Midhurst	010-659599	***NULL***
39	Bishop	D	1956	M	1980	Eaton Square	78	9629CD	Stratford	070-393435	***NULL***
44	Baker	E	1963	M	1980	Lewis Street	23	4444LJ	Inglewood	070-368753	1124
57	Brown	M	1971	M	1985	Edgecombe Way	16	4377CB	Stratford	070-473458	6409
83	Hope	PK	1956	M	1982	Magdalene Road	16a	1812UP	Stratford	070-353548	1608
95	Miller	P	1934	M	1972	High Street	33a	5746OP	Douglas	070-867564	***NULL***
100	Parmenter	P	1963	M	1979	Haseltine Lane	80	1234KK	Stratford	070-476537	6524
104	Moorman	D	1970	F	1984	Stout Street	65	9437AO	Eltham	079-987571	7060
112	Bailey	IP	1963	F	1984	Vixen Road	8	6392LK	Plymouth	010-54874	1319

Tabella **COMPANYDEL**

COMPANYID	DELIVERERID	NUMDELIVERIES	NUMCOLLECTIONS
1	2	4	8
1	6	9	1
1	8	0	1
1	44	7	5
1	57	5	0
1	83	3	3
2	8	4	4
2	27	11	2
2	104	8	4
2	112	4	8

Tabella ***COMPANIES***

COMPANYID	DELIVERERID	MANDATE
1	6	first
2	27	second

3. Query

1. Trovare per ogni fattorino che ha preso almeno due multe il codice identificativo del fattorino, la data della prima multa e la data dell'ultima multa che ha preso.
2. Trovare per ogni fattorino che ha preso almeno una multa il codice identificativo del fattorino, la data in cui ha preso l'ultima multa e l'ammontare di tale multa.
3. Trovare l'identificativo delle aziende presso cui si sono recati più del 30% dei fattorini presenti nella base di dati (nota: i fattorini "recatisi" presso un'azienda sono quelli che hanno fatto almeno una consegna o un ritiro presso l'azienda in esame).

PARTE II

Passi preliminari per lo svolgimento dell'esercitazione

La finalità di questa seconda parte dell'esercitazione è di creare, dato lo schema logico di una base di dati, appositi script di creazione e popolamento della base di dati, e di scrivere ed eseguire alcuni comandi di aggiornamento e cancellazione utilizzando il linguaggio SQL.

Questa parte dell'esercitazione utilizza MySQL, e in particolare la versione disponibile nel prodotto XAMPP.

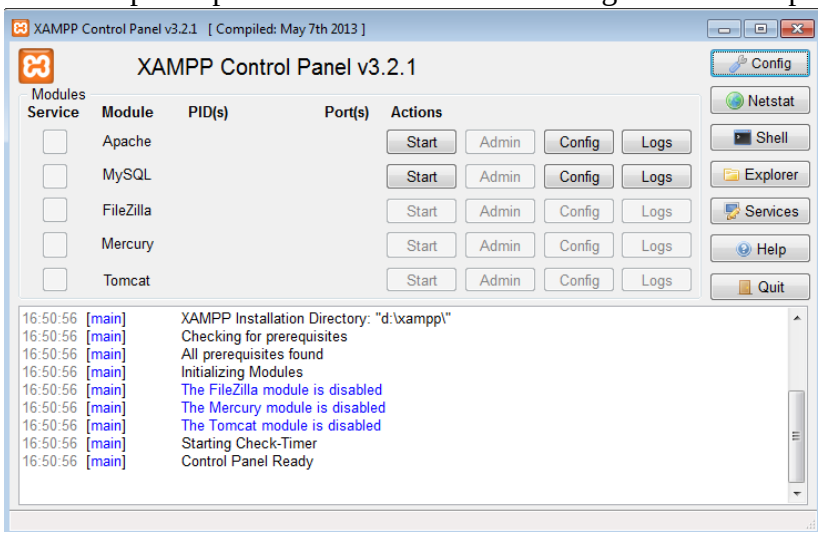
Avvio del server MySQL sulla macchina locale e avvio di Apache

L'esecuzione degli script SQL contenenti i comandi SQL per la creazione e il popolamento della base di dati avviene tramite l'interfaccia web di MySQL.
Prima di aprire l'interfaccia web di MySQL è necessario:

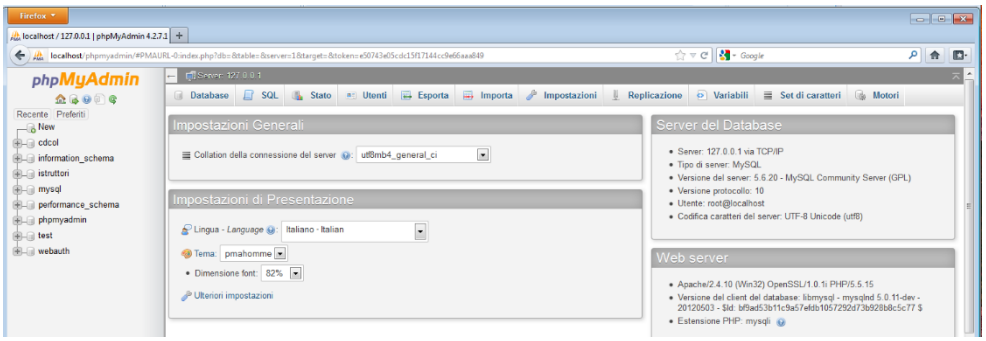
- Avviare il server locale Apache
- Avviare il server locale MySQL

In particolare, eseguire i seguenti passi:

1) Avviare il programma "XAMPP Control Panel"
2) Avviare Apache premendo il tasto Start nella riga relativa a Apache



3) Avviare MySQL premendo il tasto Start nella riga MySQL
4) Aprire l'interfaccia web di MySQL premendo il tasto Admin nella riga di MySQL (il browser si aprirà automaticamente sull'url associata alla pagina di amministrazione e interrogazione di MySQL)



5) Per eseguire uno script SQL dall'interfaccia Web di MySQL:

    a. Selezionare il pannello "*Importa*"
    b. Selezionare il file contenente lo script che si intende eseguire e poi cliccare su Esegui
6) Per rilanciare più volte lo script di creazione/popolamento ricordarsi di cancellare eventuali istanze del database creato in precedenza dal panello Database oppure includere all'inizio dello script i comandi per la cancellazione delle tabelle preesistenti

Generazione degli script di creazione e popolamento del DB

1) Gli script sono semplici file di testo scritti con un qualsiasi editor (es., Notepad, Word, Wordpad)
2) Gli script sono solitamente salvati con estensione .sql
3) Gli script contengono una sequenza di istruzioni ciascuna terminata con il simbolo ";"
4) Per creare una DB in MySQL sono necessarie le seguenti istruzioni preliminari (da scrivere all'inizio dello script):
    a. SET storage_engine=InnoDB; *(attivazione dell'Engine InnoDB per la gestione delle basi di dati)*
    b. CREATE DATABASE IF NOT EXISTS NomeDababase; *(creazione del DB denominato NomeDatabase se esso non esiste già)*
    c. USE NomeDababase; *(impostazione del DB NomeDatabase appena creato come DB corrente)*
5) Per attivare la verifica automatica del vincolo di integrità referenziale è disponibile il comando:
    a. SET FOREIGN_KEY_CHECKS=1; (*attivato*) oppure 0; (*disattivato*)
6) Alle istruzioni preliminari seguono la sequenza di istruzioni in linguaggio SQL per la creazione e il popolamento del DB (CREATE TABLE e INSERT)
7) Ricordarsi si verificare la sintassi e i tipi di dato compatibili con quelli richiesti dal DBMS MySQL
8) Se non indicato diversamente, MySQL esegue sempre il commit dopo ogni istruzione. Per la gestione delle transazioni sono disponibili i seguenti comandi:
    a. SET autocommit=0 (*disattivato*) oppure 1 (*attivato*);
    b. START TRANSACTION; (*avvio della transazione*)
    c. COMMIT; (*commit di tutte le operazioni della transazione*)

1. Descrizione della base di dati *Palestra*

La base di dati da realizzare si chiama PALESTA (usare tale nome come nome del DB) e riguarda le attività di una palestra. Essa è caratterizzata dal seguente schema logico (le chiavi primarie sono sottolineate e le chiavi esterne sono in corsivo):

ISTRUTTORE (CodFisc, Nome, Cognome, DataNascita, Email, Telefono*)

CORSI (CodC, Nome, Tipo, Livello)

PROGRAMMA (*CodFisc*, Giorno, OraInizio, Durata, *CodC*, Sala)

Per ogni istruttore è noto il codice fiscale, il nome, il cognome, la data di nascita, l'indirizzo e-mail e il numero di telefono. Per ogni corso è noto il codice, il nome, il tipo (es. attività musicale) e il livello (un numero compreso tra 1 e 4). Il programma dei corsi riporta il giorno della settimana (ad esempio lunedì, martedì, ecc.) e l'ora di inizio in cui ogni istruttore svolge una lezione di un determinato corso e la durata in minuti della lezione. Per ogni lezione programmata è noto il numero della sala in cui si svolge.

Tabella ISTRUTTORE

CodFisc	Nome	Cognome	DataNascita	Email	Telefono
SMTPLA80N31B791Z	Paul	Smith	31/12/1980	p.smith@email.it	*NULL*
KHNJHN81E30C455Y	John	Johnson	30/5/1981	j.johnson@email.it	+2300110303444
AAAGGG83E30C445A	Peter	Johnson	30/5/1981	p.johnson@email.it	+2300110303444

Tabella CORSI

CodC	Nome	Tipo	Livello
CT100	Spinning principianti	Spinning	1
CT101	Ginnastica e musica	Attività musicale	2
CT104	Spinning professionisti	Spinning	4

Tabella PROGRAMMA

CodFisc	Giorno	OrarioInizio	Durata	CodC	Sala
SMTPLA80N31B791Z	Lunedì	10:00	45	CT100	S1
SMTPLA80N31B791Z	Martedì	11:00	45	CT100	S1
SMTPLA80N31B791Z	Martedì	15:00	45	CT100	S2
KHNJHN81E30C455Y	Lunedì	10:00	30	CT101	S2
KHNJHN81E30C455Y	Lunedì	11:30	30	CT104	S2
KHNJHN81E30C455Y	Mercoledì	9:00	60	CT104	S1

Figura 1. Contenuto che deve avere la base di dati dopo l'esecuzione dello script *popolaDB.sql* creato durante l'esercitazione

2. Script

1. Creare uno script SQL (*creaDB.sql*) con le istruzioni (CREATE TABLE) per la creazione della base di dati corrispondente allo schema logico riportato nella sezione 1 del testo.

   In particolare:

   - definire tutte e tre le tabelle, scegliendo il tipo ritenuto più opportuno per gli attributi presenti nelle tabelle. Porre particolare attenzione alla definizione della chiave primaria e alla definizione dei vincoli di integrità referenziale
   - scegliere opportunamente le politiche di gestione dei vincoli di integrità referenziale selezionando quelle più idonee al contesto

   Nota. Fare attenzione all'ordine con il quale vengono create le tabelle. Creare prima le tabelle referenziate e solo in seguito quelle referenzianti.

2. Creare uno script SQL (*popolaDB.sql*) contenente le istruzioni di inserimento (INSERT) necessarie per il popolamento della base di dati creata al punto precedente. Lo script deve contenere le istruzioni di inserimento necessarie per ottenere un'istanza della base di dati contenente gli stessi dati riportati nelle tabelle presenti in figura 1.

   Nota. L'ordine di esecuzione delle INSERT è importante. Usare l'ordine corretto al fine di evitare violazioni dei vincoli di integrità referenziale.

3. Testare gli script di creazione e popolamento scritti ai punti precedenti.

   Nota. Le tre tabelle potrebbero già esistere nella base di dati nel caso in cui qualche altro studente abbia creato le tabelle durante l'orario di laboratorio precedente al vostro. In tal caso, prima di eseguire i vostri script, eseguite i seguenti comandi per cancellare le tabelle create da altri:

   - DROP TABLE PROGRAMMA;
   - DROP TABLE CORSI;
   - DROP TABLE ISTRUTTORE;

4. Scrivere in SQL e eseguire i seguenti comandi di aggiornamento, uno per volta, e verificare cosa succede nella base di dati.

   4.1. Aggiornare il numero di telefono dell'istruttore identificato dal codice fiscale 'KHNJHN81E30C455Y' impostandolo al valore '+390112333551'.

   4.2. Aggiornare la base di dati in modo tale da spostare nella sala 'S4' tutte le lezioni in programma presso la sala 'S2'.

   4.3. Eliminare dalla tabella CORSI tutti i corsi che sono in programma una sola volta a settimana (ossia che sono presenti una sola volta nella tabella PROGRAMMA). Che effetto ha l'esecuzione di questo comando sulla tabella CORSI? Che effetto ha sul contenuto della tabella PROGRAMMA? L'effetto dell'operazione sulle due tabelle è legato alla politica di gestione delle violazioni scelta durante la creazione delle tabelle?

   4.4.Eliminare l'istruttore con codice fiscale pari a 'SMTPLA80N31B791Z'. Che effetto ha questa operazione sulle tabelle ISTRUTTORE e PROGRAMMA? L'esito dell'esecuzione dell'istruzione come è legata alla politica di gestione delle violazioni imposta durante la creazione delle tabelle?

4. Soluzioni

PARTE I

1. Trovare per ogni fattorino che ha preso almeno due multe il codice identificativo del fattorino, la data della prima multa e la data dell'ultima multa che ha preso.

```
SELECT DELIVEREDID, MIN(DATA), MAX(DATA)
FROM PENALTIES
GROUP BY DELIVEREDID
HAVING COUNT(*)>=2;
```

2. Trovare per ogni fattorino che ha preso almeno una multa il codice identificativo del fattorino, la data in cui ha preso l'ultima multa e l'ammontare di tale multa.

```
SELECT P1.DELIVEREDID, DATA, AMOUNT
FROM PENALTIES P1
WHERE P1.DATA = (SELECT MAX(DATA)
                 FROM PENALTIES P2
                 WHERE P2. DELIVEREDID=P1.DELIVEREDID);
```

1. Trovare l'identificativo delle aziende presso cui si sono recati più del 30% dei fattorini presenti nella base di dati (nota: i fattorini "recatisi" presso un'azienda sono quelli che hanno fatto almeno una consegna o un ritiro presso l'azienda in esame).

```
SELECT COMPANYID
FROM COMPANYDEL
GROUP BY COMPANYID
HAVING COUNT(*) > ( SELECT 0.30*COUNT(*)
                    FROM DELIVERERS );
```

PARTE II

1. Creare uno script SQL *creaDB.sql* con le istruzioni per la creazione della base di dati corrispondente allo schema logico riportato nella Sezione 1.

```
-- create an empty database. Name of the database:
SET storage_engine=InnoDB;
SET FOREIGN_KEY_CHECKS=1;
CREATE DATABASE IF NOT EXISTS palestra;

-- use palestra
use palestra;
```

```
-- drop tables if they already exist
DROP TABLE IF EXISTS PROGRAMMA;
DROP TABLE IF EXISTS CORSI;
DROP TABLE IF EXISTS ISTRUTTORE;

-- create tables

CREATE TABLE Istruttore (
    CodFisc CHAR(20) ,
    Nome CHAR(50) NOT NULL ,
    Cognome CHAR(50) NOT NULL ,
    DataNascita DATE NOT NULL ,
    Email CHAR(50) NOT NULL ,
    Telefono CHAR(20) NULL ,
    PRIMARY KEY (CodFisc)
);

CREATE TABLE Corsi (
    CodC CHAR(10) ,
    Nome CHAR(50) NOT NULL ,
    Tipo CHAR(50) NOT NULL ,
    Livello SMALLINT NOT NULL,
    PRIMARY KEY (CodC),
    CONSTRAINT chk_Livello CHECK (Livello>=1 and Livello<=4)
);

CREATE TABLE Programma (
    CodFisc CHAR(20) NOT NULL ,
    Giorno CHAR(15) NOT NULL ,
    OraInizio TIME NOT NULL ,
    Durata SMALLINT NOT NULL ,
    Sala CHAR(5) NOT NULL,
    CodC CHAR(10) NOT NULL,
    PRIMARY KEY (CodFisc,Giorno,OraInizio),
    FOREIGN KEY (CodFisc)
     REFERENCES Istruttore(CodFisc)
     ON DELETE CASCADE
     ON UPDATE CASCADE,
    FOREIGN KEY (CodC)
     REFERENCES Corsi(CodC)
     ON DELETE CASCADE
     ON UPDATE CASCADE
);
```

2. Creare uno script SQL popolaDB.sql con le istruzioni per il popolamento della base di dati creata al punto precedente.

```
SET storage_engine=InnoDB;
SET FOREIGN_KEY_CHECKS=1;
use palestra;

-- Insert data
```

```
INSERT INTO Istruttore (CodFisc,Nome,Cognome,DataNascita,Email, Telefono)
VALUES ('SMTPLA80N31B791Z','Paul','Smith','1980-12-
31','p.smith@email.it',NULL);
INSERT INTO Istruttore (CodFisc,Nome,Cognome,DataNascita,Email, Telefono)
VALUES ('KHNJHN81E30C455Y','John','Johnson','1981-05-
30','j.johnson@email.it','+2300110303444');
INSERT INTO Istruttore (CodFisc,Nome,Cognome,DataNascita,Email, Telefono)
VALUES ('AAAGGG83E30C445A','Peter','Johnson','1981-05-
30','p.johnson@email.it','+2300110303444');
INSERT INTO Corsi (CodC,Nome,Tipo,Livello)
VALUES ('CT100','Spinning principianti','Spinning ',1);
INSERT INTO Corsi (CodC,Nome,Tipo,Livello)
VALUES ('CT101','Ginnastica e musica','Attività musicale',2);
INSERT INTO Corsi (CodC,Nome,Tipo,Livello)
VALUES ('CT104','Spinning professionisti','Spinning',4);
INSERT INTO Programma (CodFisc,Giorno,OraInizio,Durata,Sala,CodC)
VALUES ('SMTPLA80N31B791Z','Lunedì','10:00',45,'S1','CT100');
INSERT INTO Programma (CodFisc,Giorno,OraInizio,Durata,Sala,CodC)
VALUES ('SMTPLA80N31B791Z','Martedì','11:00',45,'S1','CT100');
INSERT INTO Programma (CodFisc,Giorno,OraInizio,Durata,Sala,CodC)
VALUES ('SMTPLA80N31B791Z','Martedì','15:00',45,'S2','CT100');
INSERT INTO Programma (CodFisc,Giorno,OraInizio,Durata,Sala,CodC)
VALUES ('KHNJHN81E30C455Y','Lunedì','10:00',30,'S2','CT101');
INSERT INTO Programma (CodFisc,Giorno,OraInizio,Durata,Sala,CodC)
VALUES ('KHNJHN81E30C455Y','Lunedì','11:30',30,'S2','CT104');
INSERT INTO Programma (CodFisc,Giorno,OraInizio,Durata,Sala,CodC)
VALUES ('KHNJHN81E30C455Y','Mercoledì','9:00',60,'S1','CT104');
```

3. Testare gli script di creazione e popolamento scritti ai punti precedenti

4. Parte su esecuzione dei comandi di aggiornamento (update, delete, insert)

4.1.
```
UPDATE  Istruttore
    SET Telefono = '+390112333551'
    WHERE CodFisc = 'KHNJHN81E30C455Y';
```

4.2.
```
UPDATE  Programma
    SET Sala = 'S4'
    WHERE Sala = 'S2';
```

4.3.
```
DELETE FROM Corsi
  WHERE Corsi.CodC IN (SELECT Programma.CodC
                       FROM Programma
                       GROUP BY Programma.CodC
                       HAVING COUNT(*)=1);
```

Dalla tabella Corsi viene eliminata la riga corrispondente al corso con codice CT101; dalla tabella Programma viene eliminata la riga corrispondente al corso in questione poiché era stata impostata l'opzione DELETE ON CASCASE in fase di creazione della tabella.

4.4. DELETE FROM Istruttore
WHERE CodFisc = 'SMTPLA80N31B791Z';

Grazie all'opzione DELETE ON CASCADE vengono eliminate anche le righe di Programma contenenti il codice fiscale dell'istruttore che si sta eliminando.