Linguaggio SQL

/\* file CodiceAccess.sql \*/

/\* Access SQL non supporta i commenti \*/

/\* DDL Data Definition Language \*/

/\* definizione di tabelle e relative informazioni \*/

/\* attributi

domini standard

domini definiti dall'utente (non presenti in Access ed in MySQL

vincoli intra-relazionali: constraints (NOT NULL, UNIQUE, PRIMARY KEY) \*/

/\* tipi di dati in access

INTERI bit 0-1

tinyint 0-255

smallint -32767...+32767

money 8 byte

integer INT

REALI float 8 byte

real 4 byte

decimal DEC 17 byte

STRINGA character CHAR

text

datetime

OGGETTI image \*/

/\*202\*/

CREATE TABLE materie

( ID\_materia SMALLINT NOT NULL PRIMARY KEY,

materia CHAR(20) NOT NULL);

CREATE TABLE voti

( data DATE NOT NULL,

id\_materia INT NOT NULL,

id\_alunno INT NOT NULL,

voto INT NOT NULL,

ID\_voto INT NOT NULL PRIMARY KEY);

CREATE TABLE doc

( id\_docente INT NOT NULL PRIMARY KEY,

cognome CHAR(25) NOT NULL,

nome CHAR(25) NOT NULL,

telefono CHAR(12));

CREATE TABLE alunni

( matricola SMALLINT NOT NULL PRIMARY KEY,

cognome CHAR(25),

nome CHAR(25),

telefono CHAR(12),

data\_nascita DATE NOT NULL,

UNIQUE (cognome, nome, matricola));

/\* Database gestione\_ordini \*/

/\* query crea\_clienti \*/

CREATE TABLE Clienti

( ID\_cliente INT NOT NULL,

nome CHAR (20) NOT NULL,

indirizzo CHAR(30) NOT NULL,

PRIMARY KEY(ID\_cliente));

/\* query crea\_merci \*/

CREATE TABLE Merci

( ID\_merce INT NOT NULL,

descrizione CHAR(20) NOT NULL,

prezzo FLOAT NOT NULL,

PRIMARY KEY(ID\_merce));

/\* query crea\_pagamenti \*/

CREATE TABLE Pagamenti

( ID\_pagamento INT NOT NULL,

descrizione CHAR(20) NOT NULL,

valuta CHAR(15),

PRIMARY KEY(ID\_pagamento));

/\* query crea\_ordini \*/

CREATE TABLE Ordini

( ID\_ordine INT NOT NULL,

id\_cliente INT NOT NULL,

data DATE,

id\_pagamento INT NOT NULL,

importo FLOAT,

conferma CHAR(1),

PRIMARY KEY(ID\_ordine),

FOREIGN KEY(id\_cliente) REFERENCES Clienti(ID\_cliente),

FOREIGN KEY(id\_pagamento) REFERENCES Pagamenti(ID\_pagamento));

/\* query crea\_dettagli \*/

CREATE TABLE Dettagli

(ID\_dettaglio INT NOT NULL,

id\_ordine INT NOT NULL,

id\_merce INT NOT NULL,

quantità INT NOT NULL,

PRIMARY KEY(ID\_dettaglio),

FOREIGN KEY(id\_ordine) REFERENCES Ordini(ID\_ordine),

FOREIGN KEY(id\_merce) REFERENCES Merci(ID\_merce));

/\* popolazione tabella Clienti \*/

INSERT INTO Clienti (ID\_cliente, nome, indirizzo) VALUES

(1,"Rossi","Via dei Platani,11");

INSERT INTO Clienti (ID\_cliente, nome, indirizzo) VALUES

(2,"Verdi","Via Roma,18");

INSERT INTO Clienti (ID\_cliente, nome, indirizzo) VALUES

(3,"De Antoni","Via dei Colli, 37");

INSERT INTO Clienti (ID\_cliente, nome, indirizzo) VALUES

(4,"Leone","Via Enea, 9");

/\* popolazione tabella Merci \*/

INSERT INTO Merci (ID\_merce, descrizione, prezzo) VALUES

(1,"Chiodi",4.30);

INSERT INTO Merci (ID\_merce, descrizione, prezzo) VALUES

(2,"Pinza",7.20);

INSERT INTO Merci (ID\_merce, descrizione, prezzo) VALUES

(3,"Cacciavite",2.65);

INSERT INTO Merci (ID\_merce, descrizione, prezzo) VALUES

(4,"Forbice",9.20);

/\* popolazione tabella Pagamenti \*/

INSERT INTO Pagamenti (ID\_pagamento, descrizione, valuta) VALUES

(1,"Contanti","Alla consegna");

INSERT INTO Pagamenti (ID\_pagamento, descrizione, valuta) VALUES

(2,"Assegno","Alla consegna");

INSERT INTO Pagamenti (ID\_pagamento, descrizione, valuta) VALUES

(3,"Rimessa diretta","Fine mese");

INSERT INTO Pagamenti (ID\_pagamento, descrizione, valuta) VALUES

(4,"RB 30gg fm","30 gg fine mese");

/\* popolazione tabella Ordini \*/

INSERT INTO Ordini (ID\_ordine, id\_cliente, data,

id\_pagamento, importo, conferma) VALUES

(1,1,"10/01/14",1,0,1);

INSERT INTO Ordini (ID\_ordine, id\_cliente, data,

id\_pagamento, importo, conferma) VALUES

(2,1,"12/03/14",2,0,1);

INSERT INTO Ordini (ID\_ordine, id\_cliente, data,

id\_pagamento, importo, conferma) VALUES

(3,1,"20/02/14",3,0,1);

INSERT INTO Ordini (ID\_ordine, id\_cliente, data,

id\_pagamento, importo, conferma) VALUES

(4,1,"23/01/14",4,0,1);

/\* popolazione tabella Dettagli \*/

INSERT INTO Dettagli (ID\_dettaglio, id\_ordine, id\_merce,quantità) VALUES (1,1,1,10);

INSERT INTO Dettagli (ID\_dettaglio, id\_ordine, id\_merce,quantità) VALUES (2,1,2,3);

INSERT INTO Dettagli (ID\_dettaglio, id\_ordine, id\_merce,quantità) VALUES (3,1,3,4);

INSERT INTO Dettagli (ID\_dettaglio, id\_ordine, id\_merce,quantità) VALUES (4,2,4,5);

INSERT INTO Dettagli (ID\_dettaglio, id\_ordine, id\_merce,quantità) VALUES (5,2,1,3);

INSERT INTO Dettagli (ID\_dettaglio, id\_ordine, id\_merce,quantità) VALUES (6,3,2,6);

INSERT INTO Dettagli (ID\_dettaglio, id\_ordine, id\_merce,quantità) VALUES (7,4,3,5);

/\* esempi di calcoli su funzioni di aggregazione \*/

select

(sum(peso - (select avg(peso) from Anagrafica))^2)/(select count(peso) from Anagrafica) as var,

sum(peso)/(select count(peso) from Anagrafica) as var2

from Anagrafica;

/\* creare una tabella da dati esistenti \*/

CREATE TABLE nomeTabella (campo1, campo2, . . .)

AS (SELECT Campo1, Campo2, . . .

FROM vecchiaTabella

WHERE . . . );

/\* elaborazioni su anagrafica n=1087 file anagrafica.txt \*/

/\* N ETA PESO ALTEZZA SESSO \*/

/\* presenza di dati mancanti e di outlier \*/

/\* condizionamento e pulizia dei dati \*/

/\* importa dati: da menu>Dati esterni>Importa>File di testo \*/

/\* importa in una nuova tabella, nomi nella prima riga, campi separati da spazio \*/

/\* esempio di selezione \*/

SELECT N,ETA,PESO,ALTEZZA, SESSO

FROM Anagrafica

WHERE N < 20

ORDER BY SESSO;

/\* esempio di selezione, DISTINCTROW e' ridondante \*/

SELECT DISTINCTROW

Avg([Anagrafica].[PESO]) AS [Media Di PESO],

Min([Anagrafica].[PESO]) AS [Min Di PESO],

Max([Anagrafica].[PESO]) AS [Max Di PESO],

Count(\*) AS [Conteggio Di Anagrafica]

FROM Anagrafica;

/\* crea tabella AnagPulita: tiene solo valori non nulli \*/

SELECT N, ETA, PESO, ALTEZZA, SESSO

INTO AnagPulita

FROM Anagrafica

WHERE (((Anagrafica.PESO)>0) AND ((Anagrafica.ALTEZZA)>0));

/\* verifica altri dati mancanti \*/

SELECT \* FROM AnagPulita

WHERE ISNULL(PESO);

SELECT \* FROM AnagPulita

WHERE ISNULL(ALTEZZA);

/\* aggiunge attributi PesoClassi AltClassi (la tabella non deve essere aperta) \*/

ALTER TABLE AnagPulita

ADD PesoClassi INT,

AltClassi INT;

/\* qryAltClassi uso della funzione Switch\*/

UPDATE AnagPulita

SET AltClassi = Switch(

ALTEZZA<160,1,

(ALTEZZA>=160 And ALTEZZA<170),2,

ALTEZZA>=170,3);

/\*qryPesoClassi \*/

UPDATE AnagPulita

SET PesoClassi = Switch(

PESO<60,1,

(PESO>=60 And PESO<70),2,

PESO>=70,3);

/\* Dati anomali \*/

SELECT \* FROM AnagPulita

WHERE PESO > ALTEZZA;

/\* correzione \*/

SELECT N,PESO, ALTEZZA

FROM AnagPulita

WHERE N in (298,599,1019,1026,1086);

/\* inserisce nuovi valori \*/

UPDATE AnagPulita SET

PESO =58,

ALTEZZA =179

WHERE N=298;

UPDATE AnagPulita SET

PESO =70,

ALTEZZA =161

WHERE N=599;

UPDATE AnagPulita SET

PESO =80,

ALTEZZA =130

WHERE N=1019;

UPDATE AnagPulita SET

PESO =60,

ALTEZZA =165

WHERE N=1026;

UPDATE AnagPulita SET

PESO =65,

ALTEZZA =160

WHERE N=1086;

/\* ricerca di valori anomali \*/

SELECT

MIN(PESO), MAX(PESO), MIN(ALTEZZA), MAX(ALTEZZA)

FROM AnagPulita;

SELECT \* FROM AnagPulita

WHERE PESO < 40;

SELECT COUNT(\*), PesoClassi FROM AnagPulita

GROUP BY PesoClassi;

SELECT COUNT(\*), AltClassi FROM AnagPulita

GROUP BY AltClassi;

TRANSFORM Count(N) AS ConteggioDiN

SELECT PesoClassi, Count(N) AS TotaleDiN

FROM AnagPulita

GROUP BY PesoClassi

PIVOT AltClassi;

/\* elaborazioni PesoAlt.txt (25000) file = PesoAlt25000.txt \*/

/\* importare file testo nella tabella PesoAlt \*/

/\* togliere doppi apici ai nomi campi \*/

/\* crea tabella elaborazioni \*/

SELECT

N , AltInch ,PesoPounds

INTO elaborazioni

FROM PesoAlt;

/\* aggiunge attributi PesoClassi AltClassi (la tabella non deve essere aperta) \*/

ALTER TABLE elaborazioni ADD

Altezza float,

Peso float,

AltCl int,

PesoCl int,

Bmi float,

BmiCl int);

/\* aggiunge chiave primaria \*/

ALTER TABLE elaborazioni ADD

PRIMARY KEY (N);

/\* qryImperialeMetrico \*/

UPDATE elaborazioni SET

Altezza = round((AltInch\*25.4)/10),

Peso = round((PesoPounds\*453.59237)/1000) ;

/\* qryAltClassi \*/

UPDATE elaborazioni SET AltCl = Switch(

Altezza<172,1,

(Altezza>=172 And Altezza<177),2,

Altezza>=177,3);

/\* qryPesoClassi \*/

UPDATE elaborazioni SET PesoCl = Switch(

Peso<56,1,

(Peso>=56 And Peso<63),2,

Peso>=63,3);

/\* calcolo bmi = peso kg / altezza metri ^2

qryCalcoloBmi

classi bmi <15.50 anoressia grave

15.50-17.49 anoressia moderata

17.50-18.99 leggermente sottopeso

19.00-21.99 peso ideale

22.00-24.99 sovrappeso falso magro

25.00-29.99 visibilmente sovrappeso

30.00-34.99 obesità I classe

35.00-39.99 obesità II classe

>=40.00 obesità III classe \*/

UPDATE elaborazioni SET Bmi =

round(Peso/((Altezza/100)^2),2);

UPDATE elaborazioni SET BmiCl = Switch(

Bmi < 15.5, 1,

(Bmi >= 15.50 AND Bmi < 17.50),2,

(Bmi >= 17.50 AND Bmi < 19.00),3,

(Bmi >= 19.00 AND Bmi < 22.00),4,

(Bmi >= 22.00 AND Bmi < 25.00),5,

(Bmi >= 25.00 AND Bmi < 30.00),6,

(Bmi >= 30.00 AND Bmi < 35.00),7,

(Bmi >= 35.00 AND Bmi < 40.00),8,

Bmi >= 40.00 ,9);

/\* qryCampiIncrociati peso sulle righe altezza sulle colonne \*/

TRANSFORM Count(N) AS ConteggioDiN

SELECT PesoCl, Count(N) AS TotaleDiN

FROM elaborazioni

GROUP BY PesoCl

PIVOT AltCl;

/\* verifica chi quadro con Studenti16 \*/

UPDATE Studenti16 SET AltCl = Switch(

Altezza<=167,1,

(Altezza>167 And Altezza<=175),2,

Altezza>175,3);

/\* qryPesoClassi \*/

UPDATE Studenti16 SET PesoCl = Switch(

Peso<=58,1,

(Peso>58 And Peso<=66),2,

Peso>66,3);

/\* opzionale calcolo BMI \*/

UPDATE Studenti16 SET Bmi =

round(Peso/((Altezza/100)^2),2);

UPDATE Studenti16 SET BmiCl = Switch(

Bmi < 15.5, 1,

(Bmi >= 15.50 AND Bmi < 17.50),2,

(Bmi >= 17.50 AND Bmi < 19.00),3,

(Bmi >= 19.00 AND Bmi < 22.00),4,

(Bmi >= 22.00 AND Bmi < 25.00),5,

(Bmi >= 25.00 AND Bmi < 30.00),6,

(Bmi >= 30.00 AND Bmi < 35.00),7,

(Bmi >= 35.00 AND Bmi < 40.00),8,

Bmi >= 40.00 ,9);

/\* qryCampiIncrociati tabella Pivot \*/

TRANSFORM Count(ID) AS ConteggioDiID

SELECT AltCl, Count(ID) AS TotaleDiID

FROM Studenti16

GROUP BY AltCl

PIVOT PesoCl;

/\* DDL in sql \*/

CREATE TABLE Studenti

(Uid INT UNIQUE NOT NULL PRIMARY KEY,

Cognome CHAR(30),

Nome CHAR(30));

CREATE TABLE PesoAlt

(Uid INT UNIQUE NOT NULL,

Peso INT,

Altezza INT,

PRIMARY KEY(Uid),

FOREIGN KEY(Uid) REFERENCES Studenti(Uid));

/\* alternativa al precedente \*/

CREATE TABLE Studenti

(Uid INT UNIQUE NOT NULL PRIMARY KEY,

Cognome CHAR(30),

Nome CHAR(30));

CREATE TABLE PesoAlt

(Uid INT UNIQUE NOT NULL PRIMARY KEY,

Peso INT,

Altezza INT);

ALTER TABLE PesoAlt

ADD CONSTRAINT Uid FOREIGN KEY(Uid)

REFERENCES Studenti(Uid);

/\*Access non ammette vincoli integrità referenziale in DDL SQL \*/

/\* Studenti 2 2759 studenti file Studenti2.txt \*/

/\* "n";"anno";"genere";"eta";"peso";"altezza";"dipsci";"dipcla";

"diptec";"dipalt";"componenti";"occhiali";"fumo";"diploma"

diploma 5 spazi più testo

Importare file testo nella tabella Studenti2

Verifica dei campi numerici

Chiave primaria n \*/

SELECT diploma,count(n) FROM Studenti2 GROUP BY diploma;

| **qryEsempi** | |
| --- | --- |
| **diploma** | **Expr1001** |
| alt | 211 |
| cla | 821 |
| sci | 637 |
| tec | 1090 |

SELECT

diploma,

count(n) AS frequenza,

round((count(n)/2759)\*100,2) AS percentuale

FROM Studenti2 GROUP BY diploma;

| **qryEsempi** | | |
| --- | --- | --- |
| **diploma** | **frequenza** | **percentuale** |
| alt | 211 | 7,65 |
| cla | 821 | 29,76 |
| sci | 637 | 23,09 |
| tec | 1090 | 39,51 |

/\* distribuzione univariata frequenze assolute e percentuali \*/

SELECT diploma2, count(n) AS frequenza,

(round(count(n)/(SELECT count(n) FROM Studenti2),3)\*100) AS percentuale

FROM Studenti2

GROUP BY diploma2;

| **qryEsempi** | | |
| --- | --- | --- |
| **diploma2** | **frequenza** | **percentuale** |
| alt | 211 | 7,6 |
| cla | 821 | 29,8 |
| sci | 637 | 23,1 |
| tec | 1090 | 39,5 |

/\* l’attributo diploma ha 5 spazi iniziali \*/

SELECT LEN(diploma) AS LunghezzaStringa

FROM Studenti2 WHERE n = 1;

| **qryEsempi** |
| --- |
| **LunghezzaStringa** |
| 8 |

/\* sbagliato \*/

SELECT \* FROM Studenti2 WHERE diploma='tec';

/\* corretto \*/

SELECT \* FROM Studenti2 WHERE diploma=' tec';

SELECT \* FROM Studenti2 WHERE diploma LIKE '\*tec';

SELECT n, diploma2 FROM Studenti2

WHERE diploma LIKE '\*tec' AND n<=20;

| **qryEsempi** | |
| --- | --- |
| **n** | **diploma2** |
| 1 | tec |
| 3 | tec |
| 5 | tec |
| 6 | tec |
| 12 | tec |
| 13 | tec |
| 15 | tec |
| 17 | tec |
| 18 | tec |

/\*

|  |  |  |  |
| --- | --- | --- | --- |
| Tipo di corrispondenza  operatore LIKE | Modello | Corrispondenza (viene restituito True) | Nessuna corrispondenza (viene restituito False) |
| Più caratteri | a\*a | aa, aBa, aBBBa | aBC |
|  | \*ab\* | abc, AABB, Xab | aZb, bac |
| Carattere speciale | a[\*]a | a\*a | aaa |
| Più caratteri | ab\* | abcdefg, abc | cab, aab |
| Carattere singolo | a?a | aaa, a3a, aBa | aBBBa |
| Cifra singola | a#a | a0a, a1a, a2a | aaa, a10a |
| Intervallo di caratteri | [a-z] | f, p, j | 2, & |
| All'esterno di un intervallo | [!a-z] | 9, &, % | b, a |
| Nessuna cifra | [!0-9] | A, a, &, ~ | 0, 1, 9 |
| Combinazione | a[!b-m]# | An9, az0, a99 | abc, aj0 |

\*/

ALTER TABLE Studenti2 ADD diploma2 CHAR(10);

SELECT

diploma,

RIGHT(diploma, 3) AS diploma2

FROM Studenti2;

UPDATE Studenti2

SET diploma2=RIGHT(diploma, 3) ;

/\* distribuzione dell’attributo diploma \*/

SELECT diploma2, count(diploma2) AS frequenza

FROM studenti2

GROUP BY diploma2;

/\* clausola HAVING solo su parte dell’attributo in GROUP BY \*/

SELECT diploma2, count(diploma2) AS frequenza

FROM studenti2

GROUP BY diploma2

HAVING diploma2 ='sci' OR diploma2='cla';

| **qryEsempi** |
| --- |
| **diploma2** |
| cla |
| sci |

/\* alternativa \*/

SELECT diploma2, count(diploma2) AS frequenza

FROM studenti2

GROUP BY diploma2

HAVING diploma2 IN('sci', 'cla')

/\* distribuzione bivariata diploma su righe e fuma su colonne \*/

TRANSFORM count(n)

SELECT diploma2 FROM studenti2

GROUP BY diploma2

PIVOT fumo;

| **qryEsempi** | | |
| --- | --- | --- |
| **diploma2** | **0** | **1** |
| alt | 162 | 49 |
| cla | 609 | 212 |
| sci | 472 | 165 |
| tec | 813 | 277 |

/\* selezione if \*/

SELECT IIf(fumo=0, 'non fuma', 'fuma') AS Fumatore

FROM Studenti2;

SELECT n, fumo, IIf(fumo=0, 'non fuma', 'fuma') AS Fumatore

FROM Studenti2 WHERE n<=5;

| **qryEsempi** | | |
| --- | --- | --- |
| **n** | **fumo** | **Fumatore** |
| 1 | 1 | fuma |
| 2 | 0 | non fuma |
| 3 | 0 | non fuma |
| 4 | 1 | fuma |
| 5 | 0 | non fuma |

/\* clausola BETWEEN \*/

SELECT \* FROM Studenti2 WHERE n BETWEEN 20 AND 30;

SELECT \* FROM Studenti2 WHERE n >=20 AND n <=30;

/\* clausola IN \*/

SELECT \* FROM Studenti2 WHERE n IN( 2,3,5,6,8,9);

| **qryEsempi** | | | | | | | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **n** | **anno** | **genere** | **eta** | **peso** | **altezza** | **dipsci** | **dipcla** | **diptec** | **dipalt** | **componenti** | **occhiali** | **fumo** | **diploma** | **diploma2** |
| 2 | 1988 | 0 | 20,2 | 75 | 180 | 1 | 0 | 0 | 0 | 4 | 0 | 0 | sci | sci |
| 3 | 1988 | 0 | 20,3 | 60 | 173 | 0 | 0 | 1 | 0 | 4 | 1 | 0 | tec | tec |
| 5 | 1988 | 0 | 21,4 | 66 | 164 | 0 | 0 | 1 | 0 | 5 | 0 | 0 | tec | tec |
| 6 | 1988 | 0 | 25 | 84 | 186 | 0 | 0 | 1 | 0 | 4 | 0 | 0 | tec | tec |
| 8 | 1988 | 0 | 20,6 | 89 | 170 | 0 | 1 | 0 | 0 | 3 | 1 | 0 | cla | cla |
| 9 | 1988 | 0 | 27,1 | 71 | 180 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | sci | sci |

/\* clausola GROUP BY \*/

SELECT diploma2,count(n)

FROM Studenti2

WHERE diploma2 IN('sci', 'cla')

GROUP BY diploma2;

| **qryEsempi** | |
| --- | --- |
| **diploma2** | **Expr1001** |
| cla | 821 |
| sci | 637 |

/\* proiezione su una selezione – frasi SELECT annidate \*/

SELECT n, fumo

FROM

(SELECT \*FROM studenti2 WHERE n<=5);

| **qryEsempi** | |
| --- | --- |
| **n** | **fumo** |
| 1 | 1 |
| 2 | 0 |
| 3 | 0 |
| 4 | 1 |
| 5 | 0 |

/\* calcolo della media media peso = 63,3805726712577 \*/

SELECT sum(peso)/

(SELECT count (peso) FROM Studenti2)

AS MediaPeso

FROM Studenti2;

| **qryEsempi** |
| --- |
| **MediaPeso** |
| 63,3805726712577 |

/\* verifica media peso = 63,3805726712577 \*/

SELECT round(avg(peso),4) AS MediaPeso FROM Studenti2;

/\* calcolo della varianza della popolazione – primo scarto dalla media > 65-63.38=1.62 \*/

SELECT (peso-(SELECT avg(peso) FROM Studenti2))

AS ScartiDallaMedia,

((peso-(SELECT avg(peso) FROM Studenti2))^2)

AS ScartiAlQuadrato FROM Studenti2;

| **qryEsempi** | |
| --- | --- |
| **ScartiDallaMedia** | **ScartiAlQuadrato** |
| 1,6194273287423 | 2,62254487307741 |
| 11,6194273287423 | 135,011091447923 |
| -3,3805726712577 | 11,4282715856544 |

/\* varianza peso = 133,642587421233 \*/

SELECT

sum((peso-(SELECT avg(peso) FROM Studenti2))^2)/

(SELECT count (peso) FROM Studenti2)

AS VarianzaPeso

FROM Studenti2;

| **qryEsempi** |
| --- |
| **VarianzaPeso** |
| 133,642587421233 |

/\* conferma varp=133,642587421232 /\*

SELECT varp (peso) AS VarPeso FROM Studenti2;

/\* funzione di covarianza \*/

SELECT

(peso-(SELECT avg(peso) FROM Studenti2)),

(altezza-(SELECT avg(altezza) FROM Studenti2)),

(peso-(SELECT avg(peso) FROM Studenti2))\*

(altezza-(SELECT avg(altezza) FROM Studenti2))

FROM Studenti2;

| **qryEsempi** | | |
| --- | --- | --- |
| **Expr1000** | **Expr1001** | **Expr1002** |
| 1,6194273287423 | 10,9880391446176 | 17,7943308800839 |
| 11,6194273287423 | 10,9880391446176 | 127,67472232626 |
| -3,3805726712577 | 3,9880391446176 | -13,4818561442002 |
| 29,6194273287423 | 17,9880391446176 | 532,795418230573 |

/\* covarianza = 73,3871116715654 \*/

SELECT

sum((peso-(SELECT avg(peso) FROM Studenti2))\*

(altezza-(SELECT avg(altezza) FROM Studenti2)))/

(SELECT count (peso) FROM Studenti2)

FROM Studenti2;

| **qryEsempi** |
| --- |
| **Expr1000** |
| 73,3871116715654 |

/\* correlazione = 0,770688473257721 \*/

SELECT

(sum((peso-(SELECT avg(peso) FROM Studenti2))\*

(altezza-(SELECT avg(altezza) FROM Studenti2)))/

(SELECT count (peso) FROM Studenti2))/

((SELECT stdevp(peso) FROM Studenti2)\*

(SELECT stdevp(altezza) FROM Studenti2))

FROM Studenti2;

| **qryEsempi** |
| --- |
| **Expr1000** |
| 0,770688473257721 |

/\* verifica VarPeso= 133,642587421 VarAltezza= 67,8479903195985 Cov= 73,3871116715654 \*/



SELECT

varp(peso) AS VarPeso,

varp(altezza)AS VarAlt

FROM Studenti2;

| **qryEsempi** | |
| --- | --- |
| **VarPeso** | **VarAlt** |
| 133,642587421232 | 67,8479903195985 |