Basi di dati - Laboratorio

Corso di Laurea in Bioinformatica

Docente: Barbara Oliboni

Lezione 2

Contenuto della lezione

- Vincoli interrelazionali
- Politiche di reazione alla violazione dei vincoli
- Modifica degli schemi
  - Istruzione ALTER TABLE
  - Istruzione DROP TABLE
- Operazioni sui dati
  - Istruzione INSERT
  - Istruzione DELETE
  - Istruzione UPDATE

Vincoli intrarelazionali

- Proprietà che devono essere soddisfatte da ogni istanza della base di dati
- Il soddisfacimento è definito rispetto a singole relazioni della base di dati
- Vincoli intrarelazionali di SQL (visti nella prima lezione di laboratorio):
- NOT NULL: richiede che l'attributo sia valorizzato
- UNIQUE definisce chiavi
- PRIMARY KEY: chiave primaria (c'è solo una chiave primaria in una tabella, implica NOT NULL)
- CHECK: vincolo generico

Vincoli interrelazionali

- Vincoli che coinvolgono più relazioni
- I più significativi sono i vincoli di integrità referenziale o vincoli di riferimento
  - consentono di mantenere consistenti i legami logici tra tuple
- In SQL la definizione dei vincoli di integrità referenziale si converte nella definizione di un vincolo FOREIGN KEY (chiave esterna o esportata)

FOREIGN KEY

- Crea un legame tra i valori di un attributo A (o di più attributi) della tabella corrente (*interna o Slave*) e i valori presenti nell'attributo B (o in più attributi) di un'altra tabella (*esterna o Master*)
- Impone che in ogni tupla della tabella interna il valore di A, se diverso dal valore nullo, sia presente tra i valori di B nella tabella esterna
- ATTENZIONE: L'attributo B della tabella esterna deve essere soggetto a un vincolo UNIQUE (o PRIMARY KEY). È ammesso quindi che B non sia la chiave primaria purché sia però "identificante" per le tuple della tabella esterna

FOREIGN KEY

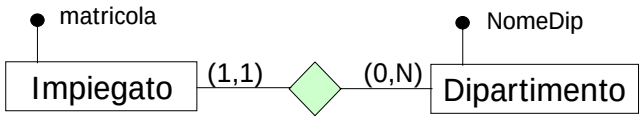
- Nel vincolo possono essere coinvolti più attributi, ad esempio quando la chiave primaria della tabella esterna è costituita da un insieme di attributi
  - Si confrontano insiemi di valori invece che singoli valori
- Un vincolo di integrità referenziale può essere definito sintatticamente in due modi:
  - Uso costrutto REFERENCES su un attributo
  - Uso costrutto FOREIGN KEY come vincolo di tabella

USO REFERENCES

- Si usa il costrutto REFERENCES quando il vincolo è definito su un solo attributo
- Con REFERENCES (nella tabella interna) si specificano
  - la tabella esterna e
  - l'attributo della tabella esterna con il quale l'attributo della tabella interna deve essere legato

Esempio schema base di dati

Schema concettuale:

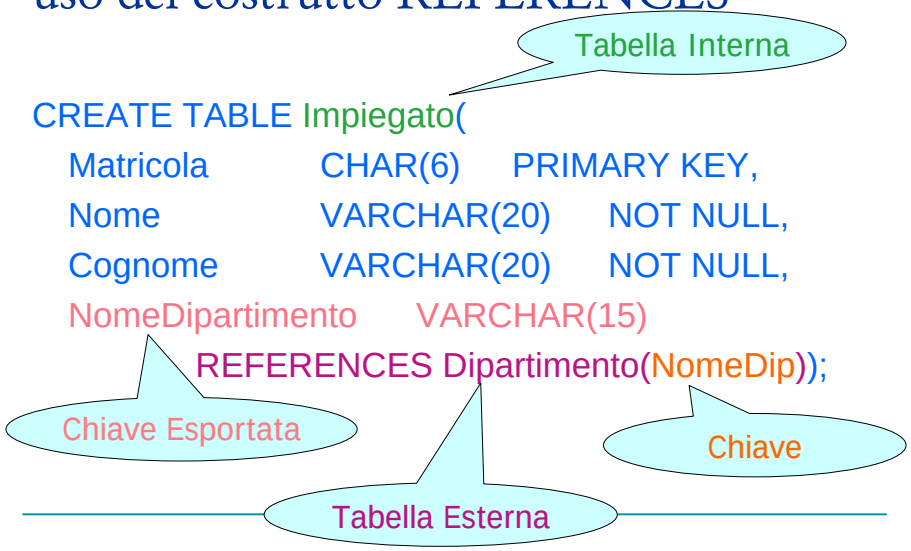


Schema relazionale:

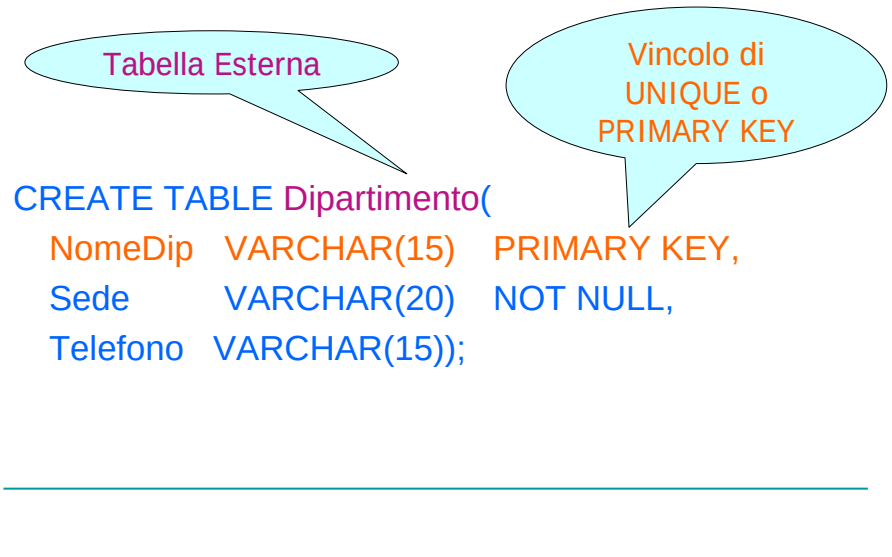
Impiegato(Matricola, Nome, Cognome, NomeDipartimento)

Dipartimento(NomeDip, Sede, Telefono)

CREATE TABLE: uso del costrutto REFERENCES



CREATE TABLE: uso del costrutto REFERENCES



ESEMPIO

Tabella Interna: IMPIEGATO

Matricola	Nome	Cognome	NomeDipartimento
A00001	Mario	Rossi	Acquisti
A00002	Paolo	Verdi	Vendite

Tabella Esterna: DIPARTIMENTO

Vincolo UNIQUE o PRIMARY KEY

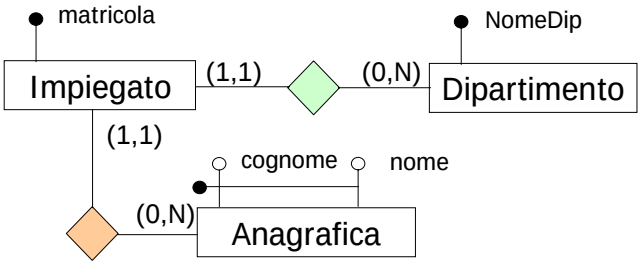
NomeDip	Sede	Telefono
Acquisti	Verona	045/8008080
Vendite	Milano	02/8007070

Costrutto FOREIGN KEY

- Si usa il costrutto FOREIGN KEY quando il vincolo di integrità referenziale è definito su un insieme di attributi
- Con FOREIGN KEY (nella tabella interna) si elencano
  - gli attributi della tabella interna coinvolti nel legame e
  - con la parola chiave REFERENCES si specificano la tabella esterna e gli attributi della tabella esterna con il quale gli attributi della tabella interna devono essere legati

Esempio schema base di dati

Schema concettuale:



Schema relazionale:

Impiegato(Matricola, Nome, Cognome, NomeDipartimento)

Dipartimento(NomeDip, Sede, Telefono)

Anagrafica(CodiceFiscale, Cognome, Nome, Indizzo)

CREATE TABLE: uso del costrutto FOREIGN KEY

Tabella Interna

```
CREATE TABLE Impiegato(
   Matricola     CHAR(6)          PRIMARY KEY,
   Nome          VARCHAR(20)      NOT NULL,
   Cognome       VARCHAR(20)      NOT NULL,
   NomeDipartimento  VARCHAR(15)
         REFERENCES Dipartimento(NomeDip),
   FOREIGN KEY(Nome,Cognome)
         REFERENCES Anagrafica(Nome,Cognome));
```

Chiave Esportata

Tabella Esterna

Attributi Chiave (ordinati)

CREATE TABLE: uso del costrutto FOREIGN KEY

Tabella Esterna

```
CREATE TABLE Anagrafica(
    CodFisc      CHAR(11)        PRIMARY KEY,
    Nome         VARCHAR(20)     NOT NULL,
    Cognome      VARCHAR(20)     NOT NULL,
    Indirizzo    VARCHAR(30),
   UNIQUE(Nome,Cognome)
);
```

Vincolo di UNIQUE

ESEMPIO

Tabella Interna: IMPIEGATO

Matricola	Nome	Cognome	NomeDipartimento
A00001	Mario	Rossi	Acquisti
A00002	Paolo	Verdi	Vendite

Tabella Esterna: ANAGRAFICA

Vincolo di UNIQUE

CodFisc	Nome	Cognome	Indirizzo
RSSMRA…	Mario	Rossi	Via X
VRDPAO…	Paolo	Verdi	Via Y

Violazione vincoli e politiche di reazione

- È possibile associare ad un vincolo di integrità referenziale una politica di reazione alle violazioni
  - SQL permette di decidere quale reazione adottare
- Per gli altri vincoli, in presenza di violazione, l'aggiornamento viene rifiutato

Violare i vincoli operando sulla tabella interna

- Si possono introdurre violazioni modificando il contenuto della tabella interna solo in due modi:
  - Modificando il valore dell'attributo referente
  - Inserendo una nuova riga
- Per queste operazioni SQL non offre nessun supporto:
  - Le operazioni vengono semplicemente impedite

ESEMPIO

Tabella Interna: IMPIEGATO

Matricola	Nome	Cognome	NomeDipartimento
A00001	Mario	Rossi	Acquisti
A00002	Paolo	Verdi	Vendite
A00003	Marco	Bianchi	Marketing

Tentativo di inserimento che causa VIOLAZIONE!!!

Tabella Esterna: DIPARTIMENTO

NomeDip	Sede	Telefono
Acquisti	Verona	045/8008080
Vendite	Milano	02/8007070

ESEMPIO

Tabella Interna: IMPIEGATO

Matricola	Nome	Cognome	NomeDipartimento
A00001	Mario	Rossi	Acquisti
A00002	Paolo	Verdi	Vendite
~~A00003~~	~~Marco~~	~~Bianchi~~	~~Marketing~~

L'inserimento viene impedito

Matricola	Nome	Cognome	NomeDipartimento
A00001	Mario	Rossi	Acquisti
A00002	Paolo	Verdi	Vendite

Violare i vincoli operando sulla tabella esterna

- Diverse alternative per rispondere a violazioni generate da modifiche sulla tabella esterna (o tabella Master)
- La tabella interna (o tabella Slave) deve adeguarsi alle modifiche che avvengono sulla tabella Master
- Le violazioni possono avvenire per:
  - Modifiche dell'attributo riferito (MA)
  - Cancellazione righe dalla tabella Master (CR)

Politiche di reazione per modifica attributo riferito

- **Cascade**: il nuovo valore dell'attributo della tabella esterna viene riportato su tutte le corrispondenti righe della tabella interna
- Esempio: modifica di un valore dell'attributo NomeDip nella tabella DIPARTIMENTO
  - DIPARTIMENTO: Da Vendite a VenditeMI
  - IMPIEGATO: Da Vendite a VenditeMI

CASCADE (MA)

Tabella Esterna: DIPARTIMENTO

NomeDip	Sede	Telefono
Acquisti	Verona	045/8008080
Vendite	Milano	02/8007070

NomeDip	Sede	Telefono
Acquisti	Verona	045/8008080
VenditeMI	Milano	02/8007070

Tabella Interna: IMPIEGATO

Matricola	Nome	Cognome	NomeDipartimento
A00001	Mario	Rossi	Acquisti
A00002	Paolo	Verdi	Vendite

Matricola	Nome	Cognome	NomeDipartimento
A00001	Mario	Rossi	Acquisti
A00002	Paolo	Verdi	VenditeMI

CASCADE (MA): risultato

NomeDip	Sede	Telefono
Acquisti	Verona	045/8008080
VenditeMI	Milano	02/8007070

Tabella Esterna: DIPARTIMENTO

Tabella Interna: IMPIEGATO

Matricola	Nome	Cognome	NomeDipartimento
A00001	Mario	Rossi	Acquisti
A00002	Paolo	Verdi	VenditeMI

Politiche di reazione per modifica attributo riferito

- **Set null**: all'attributo referente (tabella interna) viene assegnato valore nullo al posto del valore modificato nella tabella esterna
- Esempio: modifica di un valore dell'attributo NomeDip nella tabella DIPARTIMENTO
  - DIPARTIMENTO: Da Vendite a VenditeMI
  - IMPIEGATO: Da Vendite a NULL

SET NULL (MA)

Tabella Esterna: DIPARTIMENTO

NomeDip	Sede	Telefono
Acquisti	Verona	045/8008080
Vendite	Milano	02/8007070

NomeDip	Sede	Telefono
Acquisti	Verona	045/8008080
VenditeMI	Milano	02/8007070

Tabella Interna: IMPIEGATO

Matricola	Nome	Cognome	NomeDipartimento
A00001	Mario	Rossi	Acquisti
A00002	Paolo	Verdi	Vendite

Matricola	Nome	Cognome	NomeDipartimento
A00001	Mario	Rossi	Acquisti
A00002	Paolo	Verdi	NULL

Politiche di reazione per modifica attributo riferito

- **Set default**: all'attributo referente viene assegnato un valore di default al posto del valore modificato nella tabella esterna

- Esempio: modifica di un valore dell'attributo NomeDip nella tabella DIPARTIMENTO supponendo che il valore di default sia DipVendite

  - DIPARTIMENTO: Da Vendite a VenditeMI
  - IMPIEGATO: Da Vendite a DipVendite

SET DEFAULT (MA)

Tabella Esterna: DIPARTIMENTO

NomeDip	Sede	Telefono
Acquisti	Verona	045/8008080
Vendite	Milano	02/8007070

NomeDip	Sede	Telefono
Acquisti	Verona	045/8008080
VenditeMI	Milano	02/8007070

Tabella Interna: IMPIEGATO

Matricola	Nome	Cognome	NomeDipartimento
A00001	Mario	Rossi	Acquisti
A00002	Paolo	Verdi	Vendite

Matricola	Nome	Cognome	NomeDipartimento
A00001	Mario	Rossi	Acquisti
A00002	Paolo	Verdi	DipVendite

Politiche di reazione per modifica attributo riferito

- **No action**: non viene eseguita alcuna reazione. Il sistema può generare messaggio di errore ma la tabella interna non viene modificata
- Esempio: modifica di un valore dell'attributo NomeDip nella tabella DIPARTIMENTO
  - DIPARTIMENTO: Da Vendite a VenditeMI
  - IMPIEGATO: Da Vendite a Vendite

NO ACTION (MA)

Tabella Esterna: DIPARTIMENTO

Tabella Interna: IMPIEGATO

NomeDip	Sede	Telefono
Acquisti	Verona	045/8008080
Vendite	Milano	02/8007070

NomeDip	Sede	Telefono
Acquisti	Verona	045/8008080
VenditeMI	Milano	02/8007070

Matricola	Nome	Cognome	NomeDipartimento
A00001	Mario	Rossi	Acquisti
A00002	Paolo	Verdi	Vendite

Matricola	Nome	Cognome	NomeDipartimento
A00001	Mario	Rossi	Acquisti
A00002	Paolo	Verdi	Vendite

Politiche di reazione per cancellazione riga tabella esterna

- SQL mette a disposizione le stesse politiche di reazione:
  - **Cascade**: tutte le righe della tabella interna corrispondenti alla riga cancellata vengono cancellate
  - **Set null**: all'attributo referente viene assegnato il valore nullo al posto del valore presente nella riga cancellata dalla tabella esterna
  - **Set default**: all'attributo referente viene assegnato un valore di default
  - **No action**: non viene eseguita alcuna reazione

Politiche di reazione per cancellazione attributo riferito

- **Cascade**: tutte le righe della tabella interna corrispondenti alla riga cancellata vengono cancellate
- Esempio: cancellazione dalla tabella DIPARTIMENTO della riga che ha NomeDip uguale a Vendite

CASCADE (CR)

Tabella Esterna: DIPARTIMENTO

Tabella Interna: IMPIEGATO

NomeDip	Sede	Telefono
Acquisti	Verona	045/8008080
Vendite	Milano	02/8007070

NomeDip	Sede	Telefono
Acquisti	Verona	045/8008080

Matricola	Nome	Cognome	NomeDipartimento
A00001	Mario	Rossi	Acquisti
A00002	Paolo	Verdi	Vendite

Matricola	Nome	Cognome	NomeDipartimento
A00001	Mario	Rossi	Acquisti

Politiche di reazione per cancellazione attributo riferito

- **Set null**: all'attributo referente viene assegnato il valore nullo al posto del valore presente nella riga cancellata dalla tabella esterna
- Esempio: cancellazione dalla tabella DIPARTIMENTO della riga che ha NomeDip uguale a Vendite

SET NULL (CR)

Tabella Esterna: DIPARTIMENTO

Tabella Interna: IMPIEGATO

NomeDip	Sede	Telefono
Acquisti	Verona	045/8008080
Vendite	Milano	02/8007070

NomeDip	Sede	Telefono
Acquisti	Verona	045/8008080

Matricola	Nome	Cognome	NomeDipartimento
A00001	Mario	Rossi	Acquisti
A00002	Paolo	Verdi	Vendite

Matricola	Nome	Cognome	NomeDipartimento
A00001	Mario	Rossi	Acquisti
A00002	Paolo	Verdi	NULL

Politiche di reazione per cancellazione attributo riferito

- **Set default**: all'attributo referente viene assegnato valore di default
- Esempio: cancellazione dalla tabella DIPARTIMENTO della riga che ha NomeDip uguale a Vendite supponendo che il valore di default sia DipVendite

SET DEFAULT (CR)

Tabella Esterna: DIPARTIMENTO

Tabella Interna: IMPIEGATO

NomeDip	Sede	Telefono
Acquisti	Verona	045/8008080
Vendite	Milano	02/8007070

NomeDip	Sede	Telefono
Acquisti	Verona	045/8008080

Matricola	Nome	Cognome	NomeDipartimento
A00001	Mario	Rossi	Acquisti
A00002	Paolo	Verdi	Vendite

Matricola	Nome	Cognome	NomeDipartimento
A00001	Mario	Rossi	Acquisti
A00002	Paolo	Verdi	DipVendite

Politiche di reazione per cancellazione attributo riferito

- **No action**: nessuna reazione. Il sistema può generare messaggio di errore ma la tabella interna non viene modificata
- Esempio: cancellazione dalla tabella DIPARTIMENTO della riga che ha NomeDip uguale a Vendite

NO ACTION (CR)

Tabella Esterna: DIPARTIMENTO

Tabella Interna: IMPIEGATO

NomeDip	Sede	Telefono
Acquisti	Verona	045/8008080
Vendite	Milano	02/8007070

NomeDip	Sede	Telefono
Acquisti	Verona	045/8008080

Matricola	Nome	Cognome	NomeDipartimento
A00001	Mario	Rossi	Acquisti
A00002	Paolo	Verdi	Vendite

Matricola	Nome	Cognome	NomeDipartimento
A00001	Mario	Rossi	Acquisti
A00002	Paolo	Verdi	Vendite

Vincoli di integrità: sommario

- Vincoli su attributi
  - Vincolo Attributo:=
    [NOT NULL [UNIQUE]] | [CHECK (Condizione)]
    [REFERENCES Tabella [(Attributo {, Attributo})]]
    [ON {DELETE|UPDATE} {NO ACTION | CASCADE | SET NULL | SET DEFAULT}]
- Vincoli su tabella
  - Vincolo Tabella:= UNIQUE(Attributo {, Attributo})
    | CHECK(Condizione) |
    | PRIMARY KEY [Nome] Attributo {, Attributo})
    | FOREIGN KEY [Nome] Attributo {, Attributo})
    REFERENCES Tabella [(Attributo {, Attributo})]
    [ON {DELETE|UPDATE} {NO ACTION | CASCADE | SET NULL | SET DEFAULT}]

CREATE TABLE: esempio completo

```
CREATE TABLE Impiegato(
    Matricola           CHAR(6)          PRIMARY KEY,
    Nome                VARCHAR(20)      NOT NULL,
    Cognome             VARCHAR(20)      NOT NULL,
    NomeDipartimento VARCHAR(15)
          REFERENCES Dipartimento(NomeDip),
    FOREIGN KEY(Nome,Cognome)
         REFERENCES Anagrafica(Nome,Cognome)
    ON DELETE SET NULL
    ON UPDATE CASCADE);
```

Modifica degli schemi: ALTER TABLE

- Per aggiungere un nuovo attributo:

```
ALTER TABLE NomeTabella
ADD COLUMN NuovoAttr Tipo
```

- Per rimuovere un attributo:

```
ALTER TABLE NomeTabella
DROP COLUMN    NomeAttributo
```

- Per modificare il valore di default di un attributo:

```
ALTER TABLE Nometabella
ALTER COLUMN NomeAttributo
{SET DEFAULT NuovoDefault | DROP DEFAULT}
```

Modifica degli schemi: ALTER TABLE

- Per aggiungere un nuovo vincolo:

  ALTER TABLE Nometabella
  ALTER COLUMN NomeAttributo
  ADD CONSTRAINT DefVincolo

- Per rimuovere un vincolo:

  ALTER TABLE Nometabella
  ALTER COLUMN NomeAttributo
  DROP CONSTRAINT NomeVincoli

Modifica degli schemi: DROP TABLE

- Per cancellare una tabella:

  DROP TABLE NomeTabella

Modifiche degli schemi: esempi

- Aggiungere l'attributo "Stipendio" alla tabella Impiegato:

```
ALTER TABLE Impiegato
    ADD COLUMN Stipendio numeric(8,2);
```

- Per l'attributo "Stipendio" inserire come valore di default "1000.00"

```
ALTER TABLE Impiegato
    ALTER COLUMN Stipendio
    SET DEFAULT 1000.00;
```

- Rimuovere l'attributo "Stipendio" dalla tabella Impiegato:

```
ALTER TABLE Impiegato
    DROP COLUMN Stipendio;
```

- Rimuovere la tabella Impiegato:

```
DROP TABLE Impiegato;
```

SQL: operazioni sui dati (DML)

- Modifica:
  - INSERT: inserimento tuple
  - DELETE: cancellazione tuple

    ```
    DELETE FROM Tabella WHERE Condizione
    ```

  - UPDATE: aggiornamento valori tuple

    ```
    UPDATE Tabella
    SET ATTRIBUTO=Expr, …, Attributo=Expr
    WHERE Condizione
    ```

- Interrogazione:
  - SELECT <target_list>

    ```
    FROM <table_list>
    WHERE <condition>
    ```

INSERT

- Per inserire una tupla in una tabella

```
INSERT INTO NomeTabella
[(<ElencoAttributi>)]
VALUES (<Elenco di Valori>)
```

Istruzione DELETE

- Per eliminare righe dalle tabelle

```
DELETE FROM NomeTabella
[WHERE Condizione]
```

DELETE: esempio

- Eliminare le righe della tabella Dipartimento con nome del dipartimento uguale a “Vendite”

```
DELETE FROM Dipartimento
WHERE NomeDip='Vendite';
```

- Attenzione ai vincoli di integrità referenziale con politica cascade

NomeDip	Sede	Telefono
Acquisti	Verona	045/8008080
Vendite	Milano	02/8007070

Istruzione UPDATE

- Per aggiornare uno o più attributi delle righe di una tabella:

```
UPDATE NomeTabella
SET ATTRIBUTO1 = Expr,
    ATTRIBUTO2 = Expr
WHERE Condizione
```

Se la condizione non compare vengono aggiornate tutte le righe

UPDATE: esempio

- Aggiungere un dipendente nella tabella Dipartimento in corrispondenza del dipartimento Vendite

```
UPDATE Impiegato
SET Stipendio = Stipendio + 100
WHERE NomeDip='Vendite';
```

- Risultato:

Matricola	Nome	Cognome	NomeDipartimento	Stipendio
A00001	Mario	Rossi	Acquisti	1000
A00002	Paolo	Verdi	Vendite	1000 -> 1100

Istruzione SELECT

- Il comando base del linguaggio SQL:

```
SELECT Attributo {,Attributo}
FROM Tabella {,Tabella}
[WHERE Condizione]
```

SELECT: esempio 1

- Trovare tutti i dati inseriti nella tabella Impiegato

```
SELECT *
FROM Impiegato;
```

- Risultato:

Matricola	Nome	Cognome	NomeDipartimento
A00001	Mario	Rossi	Acquisti
A00002	Paolo	Verdi	Vendite

SELECT: esempio 2

- Trovare Matricola, Nome, Cognome dei dati inseriti nella tabella Impiegato

```
SELECT Matricola, Nome, Cognome
FROM Impiegato;
```

- Risultato:

Matricola	Nome	Cognome
A00001	Mario	Rossi
A00002	Paolo	Verdi

PostgreSQL

http://www.postgresql.org/

Creazione Tabella

```
dblab200=> CREATE TABLE Persona(
dblab200(> CodiceFiscale char(16) PRIMARY KEY,
dblab200(> Nome varchar(20),
dblab200(> Cognome varchar(20),
dblab200(> Indirizzo varchar(50),
dblab200(> Email varchar(30)
dblab200(> );
NOTICE:  CREATE TABLE / PRIMARY KEY will create
  implicit index "persona_pkey" for table "persona"
CREATE TABLE
dblab200=>
```

Tabella Persona

```
dblab200=>
dblab200=> \d persona
                Tabella "public.persona"
    Colonna     |          Tipo           | Modificatori
----------------+-------------------------+--------------
 codicefiscale | character(16)           | not null
 nome           | character varying(20)  |
 cognome        | character varying(20)  |
 indirizzo      | character varying(50)  |
 email          | character varying(30)  |
Indici:
    "persona_pkey" chiave primaria, btree (codicefiscale)

dblab200=>
```

Modifica Tabella
Inserimento nuovo attributo

```
dblab200=> ALTER TABLE persona
dblab200-> ADD
dblab200-> Telefono varchar(15);
ALTER TABLE
dblab200=>
```

Tabella modificata

```
dblab200=> \d Persona
               Tabella "public.persona"
    Colonna     |          Tipo          | Modificatori
----------------+------------------------+--------------
 codicefiscale | character(16)           | not null
 nome           | character varying(20)  |
 cognome        | character varying(20)  |
 indirizzo      | character varying(50)  |
 email          | character varying(30)  |
 telefono       | character varying(15)  |
Indici:
    "persona_pkey" chiave primaria, btree (codicefiscale)

dblab200=>
```

Modifica nome Tabella

```
dblab200=> \d
           Lista delle relazioni
 Schema |  Nome   |  Tipo   | Proprietario
--------+---------+---------+--------------
 public | citta   | tabella | userlab200
 public | persona | tabella | userlab200
(2 righe)
dblab200=> ALTER TABLE persona
dblab200-> RENAME TO impiegato;
ALTER TABLE
dblab200=> \d
            Lista delle relazioni
 Schema |   Nome    |  Tipo   | Proprietario
--------+-----------+---------+--------------
 public | citta     | tabella | userlab200
 public | impiegato | tabella | userlab200
(2 righe)
```

Modifica nome Attributo

```
dblab200=> ALTER TABLE Impiegato
dblab200-> RENAME COLUMN Telefono TO Num_Telefono;
ALTER TABLE
dblab200=> \d impiegato
              Tabella "public.impiegato"
    Colonna     |          Tipo           | Modificatori
----------------+-------------------------+--------------
 codicefiscale | character(16)           | not null
 nome          | character varying(20)   |
 cognome       | character varying(20)   |
 indirizzo     | character varying(50)   |
 email         | character varying(30)   |
 num_telefono  | character varying(15)   |
Indici:
    "persona_pkey" chiave primaria, btree (codicefiscale)

dblab200=>
```

Help...

```
dblab200=> \h ALTER TABLE
Comando:     ALTER TABLE
Descrizione: cambia la definizione di una tabella
Sinstassi:
ALTER TABLE [ ONLY ] nome [ * ]
    ADD [ COLUMN ] tipo colonna [ vincolo_colonna [ ... ] ]
ALTER TABLE [ ONLY ] nome [ * ]
    DROP [ COLUMN ] colonna [ RESTRICT | CASCADE ]
ALTER TABLE [ ONLY ] nome [ * ]
    ALTER [ COLUMN ] colonna { SET DEFAULT espressione | DROP DEFAULT }
ALTER TABLE [ ONLY ] nome [ * ]
    ALTER [ COLUMN ] colonna { SET | DROP } NOT NULL
ALTER TABLE [ ONLY ] nome [ * ]
    ALTER [ COLUMN ] colonna SET STATISTICS intero
ALTER TABLE [ ONLY ] nome [ * ]
    ALTER [ COLUMN ] colonna SET STORAGE { PLAIN | EXTERNAL | EXTENDED | MAIN }
ALTER TABLE [ ONLY ] nome [ * ]
    SET WITHOUT OIDS
ALTER TABLE [ ONLY ] nome [ * ]
    RENAME [ COLUMN ] colonna TO nuova_colonna
ALTER TABLE nome
    RENAME TO nuovo_nome
ALTER TABLE [ ONLY ] nome [ * ]
    ADD vincolo_tabella
ALTER TABLE [ ONLY ] nome [ * ]
    DROP CONSTRAINT nome_vincolo [ RESTRICT | CASCADE ]
ALTER TABLE nome
    OWNER TO nuovo_proprietario
ALTER TABLE nome
    CLUSTER ON nome_indice

dblab200=>
```

Eliminare Tabella

```
dblab200=> \d
            Lista delle relazioni
 Schema |   Nome     |  Tipo   | Proprietario
--------+-----------+---------+--------------
 public | citta      | tabella | userlab200
 public | impiegato  | tabella | userlab200
(2 righe)

dblab200=> DROP TABLE citta;
DROP TABLE
dblab200=> \d
            Lista delle relazioni
 Schema |   Nome     |  Tipo   | Proprietario
--------+-----------+---------+--------------
 public | impiegato  | tabella | userlab200
(1 riga)

dblab200=>
```

Vantaggio della creazione da file

```
dblab200=> \d
            Lista delle relazioni
 Schema |   Nome     |  Tipo   | Proprietario
--------+-----------+---------+--------------
 public | impiegato  | tabella | userlab200
(1 riga)

dblab200=> \i CreaCitta.txt
psql:CreaCitta.txt:1: NOTICE:  CREATE TABLE / PRIMARY KEY will
   create implicit index "citta_pkey" for table "citta"
CREATE TABLE
INSERT 2134820 1
INSERT 2134821 1
dblab200=> \d
            Lista delle relazioni
 Schema |   Nome     |  Tipo   | Proprietario
--------+-----------+---------+--------------
 public | citta      | tabella | userlab200
 public | impiegato  | tabella | userlab200
(2 righe)
```

Aggiornamento dati

```
dblab200=> SELECT * FROM impiegato;
  codicefiscale   | nome  | cognome |     indirizzo      |         email         | num_telefono
------------------+-------+---------+--------------------+-----------------------+-------------
    -
 MRARSS70D10L781T | Mario | Rossi   | Via Dante, 3, ROMA | mario.rossi@gmail.com |
(1 riga)
dblab200=> UPDATE impiegato SET num_telefono = '045 11223344'
dblab200-> WHERE codicefiscale = 'MRARSS70D10L781T';
UPDATE 1
dblab200=> SELECT * FROM impiegato;
  codicefiscale   | nome  | cognome |     indirizzo      |         email         | num_telefono
------------------+-------+---------+--------------------+-----------------------+-------------
    -
 MRARSS70D10L781T | Mario | Rossi   | Via Dante, 3, ROMA | mario.rossi@gmail.com | 045 11223344
(1 riga)
```

Eliminazione righe

```
dblab200=> SELECT * FROM citta;
 codice |  nome
--------+--------
 VR001  | Verona
 MI002  | Milano
(2 righe)

dblab200=> DELETE FROM citta
dblab200-> WHERE codice = 'MI002';
DELETE 1
dblab200=> SELECT * FROM citta;
 codice |  nome
--------+--------
 VR001  | Verona
(1 riga)

dblab200=>
```