

SQL: Amministrazione

Prof. Alfredo Pulvirenti

Prof. Salvatore Alaimo

(Atzeni-Ceri Capitoli 4 e 5)

# SQL per definire ed amministrare

- Ad ogni utente tipicamente viene associata una base di dati, creata dall'amministratore del sistema.
- L'utente diventa l'amministratore potendo stabilire gli accessi di eventuali altri utenti alla sua base di dati.
- La creazione consiste nel definire incrementalmente uno schema con un certo nome, interattivamente o da programma, e tutti i suoi elementi vengono registrati in un catalogo.

# CREATE SCHEMA

CREATE SCHEMA *Nome* AUTHORIZATION *Utente*

## *Definizioni*

- Dove:
  - Si crea il database chiamato *Nome*
  - *Utente* e' l'amministratore
  - Le *Definizioni* creano gli elementi dello schema (Tabelle, Viste, Indici, etc...)

# DROP SCHEMA

**DROP SCHEMA *Nome* [RESTRICT | CASCADE]**

- cancella la base di dati *Nome*
- Restrict: drop non viene eseguito se il database non è vuoto.
- Cascade: Vengono rimossi automaticamente tutti i dati presenti nel database

# Definizione dei dati in SQL

- Istruzione **CREATE TABLE**:
  - definisce uno schema di relazione e ne crea un'istanza vuota
  - specifica attributi, domini e vincoli

# Create Table

```
CREATE TABLE Nome (  
    Attributo Tipo[Vincolo {,Vincolo}]  
    {,Attributo Tipo[Vincolo {,Vincolo}]}  
  
    [,VincoloDiTabella {, VincoloDiTabella}]  
)
```

# CREATE TABLE, esempio

```
CREATE TABLE Impiegato(  
    Matricola CHAR(6) PRIMARY KEY,  
    Nome CHAR(20) NOT NULL,  
    Cognome CHAR(20) NOT NULL,  
    Dipart CHAR(15),  
    Stipendio NUMERIC(9) DEFAULT 0,  
    FOREIGN KEY(Dipart) REFERENCES  
    Dipartimento(NomeDip),  
    UNIQUE (Cognome, Nome)  
)
```

- Domini elementari (predefiniti)
- Domini definiti dall'utente (semplici, ma riutilizzabili)



# Domini elementari

- **Carattere**: singoli caratteri o stringhe, anche di lunghezza variabile
- **Numerici**, esatti e approssimati
- **Data, ora**
- Sistemi diversi estendono il set di base con domini non standard (vettori, periodi, ecc.)

# Domini o tipi

- **CHAR(n)** stringhe di lunghezza n
- **VARCHAR(n)** stringhe di lunghezza variabile con al massimo n caratteri
- **INTEGER** interi
- **REAL** reali
- **NUMERIC (p,s)** p cifre di cui s decimali
- **FLOAT(p)** es. 0.17E16
- **DATE, TIME** per date ed ore.

# Definizione di domini

- Istruzione **CREATE DOMAIN**:
  - definisce un dominio (semplice), utilizzabile in definizioni di relazioni, anche con vincoli e valori di default

# CREATE DOMAIN, esempio

```
CREATE DOMAIN Voto  
AS SMALLINT DEFAULT NULL  
CHECK ( value >=18 AND value <= 30 )
```

# Vincoli d'integrità

- Riguardano i valori ammissibili degli attributi di una tupla
  - **Vincoli Intrarelazionali:** nell'ambito della stessa relazione
  - **Vincoli Referenziali (o Interrelazionali):** tra diverse relazioni
- Vengono controllati durante le tre possibili operazioni di modifica SQL
  - INSERT,DELETE e UPDATE
  - Devono essere sempre soddisfatti altrimenti la transazione fallisce
  - Oppure, l'utente può opzionalmente definire delle azioni (correttive) da intraprendere per ripristinare l'integrità

## A cosa servono?

- Migliorare la *qualità* dei dati
- Arricchire semanticamente la base di dati
- La loro definizione è parte del processo di progettazione del data base
- Usati internamente dal sistema per ottimizzare l'esecuzione

# Vincoli intrarelazionali

- NOT NULL
- UNIQUE definisce chiavi
- PRIMARY KEY: chiave primaria (una sola, implica NOT NULL)
- CHECK, vedremo più avanti

# UNIQUE e PRIMARY KEY

- due forme:
  - nella definizione di un attributo, se forma da solo la chiave
  - come elemento separato



# CREATE TABLE, esempio

```
CREATE TABLE Impiegato(  
    Matricola CHAR(6) PRIMARY KEY,  
    Nome CHAR(20) NOT NULL,  
    Cognome CHAR(20) NOT NULL,  
    Dipart CHAR(15),  
    Stipendio NUMERIC(9) DEFAULT 0,  
    FOREIGN KEY(Dipart) REFERENCES  
        Dipartimento(NomeDip),  
    UNIQUE (Cognome,Nome)  
)
```

# PRIMARY KEY, alternative

Matricola CHAR(6) PRIMARY KEY

Matricola CHAR(6),

...,

PRIMARY KEY (Matricola)

# CREATE TABLE, esempio

```
CREATE TABLE Impiegato(  
    Matricola CHAR(6) PRIMARY KEY,  
    Nome CHAR(20) NOT NULL,  
    Cognome CHAR(20) NOT NULL,  
    Dipart CHAR(15),  
    Stipendio NUMERIC(9) DEFAULT 0,  
    FOREIGN KEY(Dipart) REFERENCES  
        Dipartimento(NomeDip),  
    UNIQUE (Cognome,Nome)  
)
```

# Chiavi su più attributi, attenzione

Nome CHAR(20) NOT NULL,  
Cognome CHAR(20) NOT NULL,  
UNIQUE (Cognome, Nome),

Nome CHAR(20) NOT NULL UNIQUE,  
Cognome CHAR(20) NOT NULL UNIQUE,

- Non è la stessa cosa!

# Vincoli interrelazionali

- **REFERENCES** e **FOREIGN KEY** permettono di definire vincoli di integrità referenziale
- di nuovo due sintassi
  - per singoli attributi
  - su più attributi
- E' possibile definire politiche di reazione alla violazione

**CHECK**, vedremo più avanti

<u>Codice</u>	Data	Vigile	Prov	Numero
34321	1/2/95	3987	MI	39548K
53524	4/3/95	3295	TO	E39548
64521	5/4/96	3295	PR	839548
73321	5/2/98	9345	PR	839548

## Vigili

<u>Matricola</u>	Cognome	Nome
3987	Rossi	Luca
3295	Neri	Piero
9345	Neri	Mario
7543	Mori	Gino

<u>Codice</u>	Data	Vigile	Prov	Numero
34321	1/2/95	3987	MI	39548K
53524	4/3/95	3295	TO	E39548
64521	5/4/96	3295	PR	839548
73321	5/2/98	9345	PR	839548

**Auto**

<u>Prov</u>	<u>Numero</u>	Cognome	Nome
MI	39548K	Rossi	Mario
TO	E39548	Rossi	Mario
PR	839548	Neri	Luca

# CREATE TABLE, esempio

```
CREATE TABLE Infrazioni(  
  Codice CHAR(6) NOT NULL PRIMARY KEY,  
  Data DATE NOT NULL,  
  vigile INTEGER NOT NULL  
    REFERENCES vigili(Matricola),  
  Provincia CHAR(2),  
  Numero CHAR(6) ,  
  FOREIGN KEY(Provincia, Numero)  
  REFERENCES Auto(Provincia, Numero)  
)
```



## Esempio di DB semanticamente errato

Esami	Studente	Voto	Lode	Corso
	276545	32		01
	276545	30	e lode	02
	787643	27	e lode	03
	739430	24		04

Studenti	Matricola	Cognome	Nome
	276545	Rossi	Mario
	787643	Neri	Piero
	787643	Bianchi	Luca

# Vincoli su una n-upla

- **NOT NULL :**
  - e' implicito se l'attributo fa parte di una chiave primaria
- **CHECK Condizione :**
  - specifica i valori ammissibili; esempio: Voto INTEGER NOT NULL CHECK (18 <= Voto AND Voto <= 31)
- **DEFAULT(Costante | NULL)**
  - assegna quel valore di default per ogni inserimento
- **CHECK Condizione :**
  - anche per attributi **diversi** della **stessa** n-upla

# Vincoli Intrarelazionali

- **UNIQUE:**
  - l'attributo e' una chiave
- **PRIMARY KEY [Nome Chiave]**  
**“(”Attributo{,Attributo} “)”**
  - dove gli attributi devono essere dichiarati tutti NOT NULL
- **UNIQUE “(”Attributo{,Attributo} “)”**
  - definisce una chiave con piu' attributi

# Vincoli d'integrita' referenziali

- Tuple di relazioni diverse sono correlati per mezzo del valore di chiavi (primarie)
- Servono a garantire che i valori in una certa tabella facciano riferimento a valori reali di un'altra tabella

```
FOREIGN KEY [NomeChiaveEsterna]  
“(”Attributo{,Attributo} “)”  
REFERENCES TabellaRef  
ON DELETE {NO ACTION,CASCADE,SET NULL}
```

- dove per la TabellaRef e' stata definita una chiave primaria.
- Impedisce l'inserzione di n-uple con il valore della chiave esterna che non corrisponde ad un valore della chiave primaria della TabellaRef . Se un'operazione di cancellazione su TabellaRef viola il vincolo referenziale allora vengono applicate le tre azioni:

# Azioni per Vincoli su Chiavi Esterne

- **ON DELETE NO ACTION :**
  - rifiuta l'operazione (la piu' diffusa nei DBMS)
- **ON DELETE CASCADE :**
  - cancella tutte le n-uple con valori della chiave esterna corrispondenti alla chiave primaria delle n-uple cancellate
- **ON DELETE SET NULL**
  - assegna il valore NULL agli attributi della chiave esterna

# Rifiuto della cancellazione

<u>Matricola</u>	Cognome	Progetto
34321	Rossi	IDEA
53524	Neri	XYZ
64521	Verdi	NULL
73032	Bianchi	IDEA

## Progetti

<u>Codice</u>	Inizio	Durata	Costo
IDEA	01/2000	36	200
XYZ	07/2001	24	120
BOH	09/2001	24	150

- La transazione **fallisce** e XYZ non può essere cancellato dalla relazione Progetti

# Eliminazione in cascata

<u>Matricola</u>	Cognome	Progetto
34321	Rossi	IDEA
53524	Neri	XYZ
64521	Verdi	NULL
73032	Bianchi	IDEA

## Progetti

<u>Codice</u>	Inizio	Durata	Costo
IDEA	01/2000	36	200
XYZ	07/2001	24	120
BOH	09/2001	24	150

- La transazione termina e XYZ viene cancellato anche dalla relazione Impiegati



# Introduzione di valori nulli

<u>Matricola</u>	Cognome	Progetto
34321	Rossi	IDEA
53524	Neri	NULL
64521	Verdi	NULL
73032	Bianchi	IDEA

## Progetti

<u>Codice</u>	Inizio	Durata	Costo
IDEA	01/2000	36	200
XYZ	07/2001	24	120
BOH	09/2001	24	150

- La transazione termina e all'attributo Impiegati.Progetto viene assegnato NULL

# Esempio Riassuntivo

- CREATE TABLE Clienti (  
    CodiceCliente CHAR(3) UNIQUE NOT NULL,  
    Nome CHAR(30) NOT NULL,  
    Citta' CHAR(30) NOT NULL,  
    Sconto INTEGER NOT NULL  
        CHECK(Sconto>0 AND Sconto<100),  
    PRIMARY KEY pk\_Clienti(CodiceCliente))
- CREATE TABLE Agenti (  
    CodiceAgente CHAR(3) UNIQUE NOT NULL,  
    Nome CHAR(30) NOT NULL,  
    Zona CHAR(8) NOT NULL,  
    Supervisore CHAR(3),  
    Commissione INTEGER)  
    PRIMARY KEY pk\_Agenti(CodiceAgente),  
    CHECK (Supervisore = CodiceAgente OR Supervisore IS NULL)

# Esempio Riassuntivo

- ```
CREATE TABLE Ordini(  
    NumOrdine CHAR(3) NOT NULL,  
    CodiceCliente CHAR(3) NOT NULL,  
    CodiceAgente CHAR(3) NOT NULL,  
    Data CHAR(8) NOT NULL,  
    Prodotto CHAR(3) NOT NULL,  
    Ammontare INTEGER NOT NULL CHECK (Ammontare > 100)  
    PRIMARY KEY pk-Ordini (NumOrdine)  
    FOREIGN KEY fk_ClienteOrdine (CodiceCliente)  
        REFERENCES Clienti ON DELETE NO ACTION  
    FOREIGN KEY fk_AgenteOrdine (CodiceAgente)  
        REFERENCES Agenti ON DELETE NO ACTION
```

# Modifiche degli schemi

ALTER DOMAIN

ALTER TABLE

DROP DOMAIN

DROP TABLE

...

# SQL, operazioni sui dati

- interrogazione:
  - SELECT
- modifica:
  - INSERT, DELETE, UPDATE

# Insert

```
INSERT INTO Tabella [“(”Attributo  
{,Attributo} “)”] VALUES “(” Valore  
{,Valore} “)”
```

- Esempio
  - INSERT INTO Esami VALUES ('DB1', 123456, 27)

## DELETE FROM Tabella WHERE Condizione

- Esempio:
  - DELETE FROM Esami WHERE Matricola = 123456

UPDATE Tabella SET Attributo = Espr  
{,Attributo = Espr} WHERE Condizione

- Esempio:
  - UPDATE Aule SET Aula = 126 WHERE Aula = 3



```
CREATE TABLE Studenti(  
  Nome CHAR(30),  
  Matricola INTEGER,  
  Indirizzo CHAR(30),  
  Telefono INTEGER)
```

```
CREATE TABLE FuoriCorso LIKE Studenti
```

CREATE TABLE Nome AS EsprSelect

Esempio:

CREATE TABLE EsamiBuoni

LIKE Esami AS SELECT \*

FROM Esami

WHERE Voto > 27