

# BASI DI DATI I

- Structured Query Language
- Data Definition Language - SQL



# **STRUCTURED QUERY LANGUAGE (SQL)**



# SQL

- Il linguaggio SQL permette la **definizione**, la **manipolazione** (aggiornamento e recupero) e la **gestione** di basi di dati relazionali.
- È una delle ragioni del successo dei db relazionali in ambito commerciale:  
*essendo uno standard in tutti i DBMS relazionali, gli utenti sono poco propensi a migrare verso data model diversi, quali il gerarchico o il reticolare.*



# SQL – I VANTAGGI DI UNO STANDARD (1)

- SQL è considerato la principale ragione di successo dei database relazionali. Infatti:
  - È uno standard – è possibile migrare da un DBMS relazionale verso un altro DBMS relazionale senza eccessivi problemi.
  - I programmi applicativi usano lo stesso insieme di istruzioni SQL per accedere a DBMS relazionali diversi, senza dover cambiare i sottolinguaggi.
- *I costi di formazione del personale sono ridotti*: la formazione è concentrata su di un solo linguaggio.
- *Produttività*: i tecnici, una volta imparato il linguaggio (e poiché usano solo questo) diventano sempre più esperti e produttivi.
- *Portabilità delle applicazioni*: le applicazioni possono essere spostate da una macchina all'altra, se entrambe usano SQL.



# SQL – I VANTAGGI DI UNO STANDARD (2)

- *Longevità delle applicazioni:* uno standard tende a rimanere in voga per diverso tempo e non c'è necessità di riscrivere le applicazioni.
- *Ridotta dipendenza da un singolo produttore:* quando non è usato un linguaggio proprietario, è più facile usare differenti produttori di DBMS.

Di conseguenza il mercato sarà più competitivo ed i prezzi calano.

- *Comunicazione fra sistemi:* differenti DBMS e programmi applicativi possono comunicare più facilmente.



# **MA HA ANCHE DEGLI SVANTAGGI...**

- Può in qualche modo limitare la creatività e l'innovazione.
- Può non incontrare tutte le necessità di un'industria.
- Può essere difficile da cambiare, poiché molti produttori hanno investito su di esso.
- L'utilizzo di speciali funzionalità aggiunte a SQL da qualche produttore può far perdere i vantaggi di cui alle slide precedenti.



# SQL: UN PO' DI STORIA

- Nel 1970 Codd propone il modello relazionale: iniziano esperimenti e ricerche per la realizzazione di linguaggi relazionali, cioè di linguaggi in grado di realizzare le caratteristiche del modello astratto.
- Il primo risultato è ***SEQUEL*** (*Structured English QUery Language*), definito all'IBM Research:
  - Facile da imparare e utilizzare;
  - Basato su termini inglesi che mascherano i difficili concetti **dell'algebra relazionale**.



# SQL: UN PO' DI STORIA (2)

- Una versione rivista, il **SEQUEL/2**, ridenominata **SQL** (**S**tructured **Q**uery **L**anguage) viene definita nel 1976.
- Il primo prodotto basato su SQL viene chiamato **Oracle** (1979), lanciato dalla Relational Software, Inc.
- Nel 1981 IBM annuncia un prodotto SQL denominato **SQL/Data System**; nel 1983 viene rilasciato il DBMS relazionale **DB2** compatibile con SQL/DS.



# SQL: UN PO' DI STORIA (3)

- Oggi SQL è implementato da tutti i principali fornitori di DBMS, ed è il linguaggio per database più usato al mondo.
- L'ANSI e l'ISO hanno sviluppato una serie di standard per SQL:
  - ANSI SQL-86, SQL-92 (SQL2) ed SQL3.
  - Sfortunatamente ogni DBMS relazionale implementa un suo livello (o **dialetto**) di SQL, che è un'estensione o un sottoinsieme di un livello standard.



# IL LINGUAGGIO SQL

- SQL fornisce *statement* per la definizione di dati, query e aggiornamenti, quindi è sia un **DDL** che un **DML**.
- Fornisce inoltre facility per definire viste e per ricavare indici.
- SQL può essere usato **interattivamente** (con maschere del DBMS) o essere **incorporato (embedded)** in programmi C, Cobol, Java, etc.



# **DEFINIZIONE DI DATI, SCHEMI E VINCOLI IN SQL**



# TERMINOLOGIA SQL

- SQL ha una terminologia diversa da quella classica relazionale:
  - Relazione → Tabella
  - Tupla → Riga
  - Attributo → Colonna



# CONCETTI DI SCHEMA

- Il concetto di **schema** SQL è usato per raggruppare tavelle ed altri costrutti che appartengono alla stessa applicazione di database.
- Uno schema SQL è identificato da un **nome dello schema**, ed include un identificatore di autorizzazione per indicare l'utente proprietario dello schema, così come dei **descrittori** per ogni elemento dello schema.



# CONCETTO DI SCHEMA (2)

- Uno schema include:
  - Tabelle
  - Domini
  - Viste
  - Altri costrutti, quali permessi di autorizzazione, etc.
- La sintassi per creare uno schema è

**CREATE SCHEMA** *nome\_schema* **AUTHORIZATION** *nome\_utente*

- Crea uno schema chiamato *nome\_schema*, il cui proprietario è l'utente con account *nome\_utente*.



# IL COMANDO CREATE TABLE

- **CREATE TABLE** è usato per specificare una nuova relazione, assegnandole un **nome** ed un **insieme di attributi e vincoli**.
- Gli attributi sono specificati da un **nome**, un **tipo di dato** per definire il dominio dei valori, ed eventuali **vincoli**.
- In ultimo si specifica la chiave, i vincoli di integrità di entità e di integrità referenziale.



# ISTANZA DI DATABASE RELAZIONALE

EMPLOYEE	FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO
John		Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5	
Franklin		Wong	333445555	1955-12-08	638 Vass, Houston, TX	M	40000	888665555	5	
Alicia		Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4	
Jennifer		Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4	
Ramesh		Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5	
Joyce		English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5	
Ahmad		Jabber	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4	
James		Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	null	1	

DEPARTMENT	DNAME	DNUMBER	MGRSSN	MGRSTARTDATE	DEPT_LOCATIONS	DNUMBER	DLOCATION
Research	5	333445555	1968-05-22		Houston		
Administration	4	987654321	1995-01-01		Stafford		
Headquarters	1	888665555	1981-06-19		Bellaire		
					Sugarland		

WORKS_ON	ESSN	PNO	HOURS
123456789	1	32.5	
123456789	2	7.5	
666884444	3	40.0	
453453453	1	20.0	
453453453	2	20.0	
333445555	2	10.0	
333445555	3	10.0	
333445555	10	10.0	
333445555	20	10.0	
999887777	30	30.0	
999887777	10	10.0	
987987987	10	35.0	
987987987	30	5.0	
987654321	30	20.0	
987654321	20	15.0	
888665555	20	null	

PROJECT	PNAME	PNUMBER	PLOCATION	DNUM
ProductX	1	Bellaire	5	
ProductY	2	Sugarland	5	
ProductZ	3	Houston	5	
Computerization	10	Stafford	4	
Reorganization	20	Houston	1	
Newbenefits	30	Stafford	4	

DEPENDENT	ESSN	DEPENDENT_NAME	SEX	BDATE	RELATIONSHIP
333445555		Alice	F	1986-04-05	DAUGHTER
333445555		Theodore	M	1983-10-25	SON
333445555		Joy	F	1958-05-03	SPOUSE
987654321		Abner	M	1942-02-28	SPOUSE
123456789		Michael	M	1988-01-04	SON
123456789		Alice	F	1988-12-30	DAUGHTER
123456789		Elizabeth	F	1967-05-05	SPOUSE

Un'istanza del database "Company"



# CREATE TABLE: ESEMPIO

```
CREATE TABLE EMPLOYEE
  ( FNAME          VARCHAR(15)      NOT NULL,
    MINIT          CHAR ,
    LNAME          VARCHAR(15)      NOT NULL,
    SSN            CHAR(9)         NOT NULL,
    BDATE          DATE ,
    ADDRESS        VARCHAR(30) ,
    SEX             CHAR ,
    SALARY         DECIMAL(10,2),
    SUPERSSN       CHAR(9) ,
    DNO             INT             NOT NULL,
    PRIMARY KEY (SSN),
    FOREIGN KEY (SUPERSSN) REFERENCES EMPLOYEE(SSN),
    FOREIGN KEY (DNO) REFERENCES DEPARTMENT(DNUMBER));
CREATE TABLE DEPARTMENT
  ( DNAME          VARCHAR(15)      NOT NULL,
    DNUMBER         INT             NOT NULL,
    MGRSSN         CHAR(9)         NOT NULL,
    MGRSTARTDATE   DATE ,
    PRIMARY KEY (DNUMBER),
    UNIQUE (DNAME),
    FOREIGN KEY (MGRSSN) REFERENCES EMPLOYEE(SSN));
CREATE TABLE DEPT_LOCATIONS
  ( DNUMBER         INT             NOT NULL,
    DLOCATION       VARCHAR(15)      NOT NULL,
    PRIMARY KEY (DNUMBER, DLOCATION),
    FOREIGN KEY (DNUMBER) REFERENCES DEPARTMENT(DNUMBER));
CREATE TABLE PROJECT
  ( PNAME          VARCHAR(15)      NOT NULL,
    PNUMBER         INT             NOT NULL,
    PLOCATION       VARCHAR(15) ,
    DNUM            INT             NOT NULL,
    PRIMARY KEY (PNUMBER),
    UNIQUE (PNAME),
    FOREIGN KEY (DNUM) REFERENCES DEPARTMENT(DNUMBER));
CREATE TABLE WORKS_ON
  ( ESSN           CHAR(9)         NOT NULL,
    PNO             INT             NOT NULL,
    HOURS          DECIMAL(3,1)    NOT NULL,
    PRIMARY KEY (ESSN, PNO),
    FOREIGN KEY (ESSN) REFERENCES EMPLOYEE(SSN),
    FOREIGN KEY (PNO) REFERENCES PROJECT(PNUMBER));
CREATE TABLE DEPENDENT
  ( ESSN           CHAR(9)         NOT NULL,
    DEPENDENT_NAME VARCHAR(15)    NOT NULL,
    SEX              CHAR ,
    BDATE           DATE ,
    RELATIONSHIP    VARCHAR(8) ,
    PRIMARY KEY (ESSN, DEPENDENT_NAME),
    FOREIGN KEY (ESSN) REFERENCES EMPLOYEE(SSN));
```

Gli statement SQL2 per definire lo schema del database "Company"

# TIPI DI DATI E DOMINI

- Tipi di dati disponibili in SQL2:
  - Numerici
    - Interi (INTEGER o INT, SMALLINT)
    - Reali (FLOAT, REAL, DOUBLE PRECISION)
    - Numeri formattati (DECIMAL(i,j), DEC(i,j), NUMERIC(i,j))
      - i, detta precisione, indica il numero di cifre decimali.
      - j, detta scala, indica il numero di cifre dopo la virgola.
  - Stringhe di caratteri
    - A lunghezza fissa (CHAR(n), CHARACTER(n))
    - A lunghezza variabile (VARCHAR(n) o CHAR VARYING(n))
      - Per default n, il numero massimo di caratteri, è 1.



# TIPI DI DATI E DOMINI (2)

- **Stringhe di bit:**

- A lunghezza fissa (BIT(n))
- A lunghezza variabile (BIT VARYING(n))

- **DATE:**

- Ha dieci posizioni, con componenti YEAR, MONTH e DAY.
- Formato *YYYY-MM-DD*.

- **TIME:**

- Ha (almeno) otto posizioni con componenti HOUR, MINUTE e SECOND.
- Formato *HH:MM:SS*.



# **DOMINI PERSONALIZZATI**



# I VALORI NULL E DEFAULT

- Poiché SQL consente che un attributo abbia valore **null**, se si vuole impedire ciò si usa il vincolo  
**NOT NULL**.
  - Tale vincolo deve sempre essere specificato per la chiave primaria (*vincolo di integrità di entità*).
- È anche possibile specificare un valore di default per un attributo, attraverso la clausola  
**DEFAULT <value>**, dopo la dichiarazione dell'attributo
  - Senza tale clausola il valore di default di un attributo è null.



# ALTRI VINCOLI

- Dopo le specifiche degli attributi, possono essere specificati i **vincoli di tabella**, quali **chiave** ed **integrità referenziale**:
  - La clausola **PRIMARY KEY** specifica uno o più attributi che faranno da chiave primaria.
  - La clausola **UNIQUE** specifica una chiave alternativa.
  - La clausola **FOREIGN KEY** specifica l'integrità referenziale.



## ALTRI VINCOLI (2)

- Il progettista dello schema può specificare l'azione da intraprendere se si viola un vincolo di integrità referenziale, attraverso la cancellazione di una tupla referenziata o attraverso la modifica di un valore di chiave referenziata.
- L'azione referenziale **triggered** può essere specificata nella clausola **FOREIGN KEY**.
  - Possibili azioni sono **SET NULL**, **CASCADE** e **SET DEFAULT**, qualificate da opzioni **ON DELETE** e **ON UPDATE**.



# ALTRI VINCOLI: ESEMPIO

```
CREATE TABLE EMPLOYEE
(
    ...,
    DNO          INT  NOT NULL  DEFAULT 1,
    CONSTRAINT EMPPK
        PRIMARY KEY (SSN),
    CONSTRAINT EMPSUPERFK
        FOREIGN KEY (SUPERSSN) REFERENCES EMPLOYEE(SSN)
            ON DELETE SET NULL  ON UPDATE CASCADE ,
    CONSTRAINT EMPDEPTFK
        FOREIGN KEY (DNO) REFERENCES DEPARTMENT(DNUMBER)
            ON DELETE SET DEFAULT  ON UPDATE CASCADE );
```

```
CREATE TABLE DEPARTMENT
(
    ...,
    MGRSSN  CHAR(9) NOT NULL DEFAULT '888665555',
    ...,
    CONSTRAINT DEPTPK
        PRIMARY KEY (DNUMBER),
    CONSTRAINT DEPTSK
        UNIQUE (DNAME),
    CONSTRAINT DEPTMGRFK
        FOREIGN KEY (MGRSSN) REFERENCES EMPLOYEE(SSN)
            ON DELETE SET DEFAULT  ON UPDATE CASCADE );
```

```
CREATE TABLE DEPT_LOCATIONS
(
    ...,
    PRIMARY KEY (DNUMBER, DLOCATION),
    FOREIGN KEY (DNUMBER) REFERENCES DEPARTMENT(DNUMBER)
        ON DELETE CASCADE  ON UPDATE CASCADE );
```

Specifica di valori di default e azioni referenziali triggered.



# ALTRI VINCOLI: ESEMPIO

- Nell'esempio, per la chiave esterna SUPERSSN di EMPLOYEE ci sono i vincoli:
  - **SET NULL ON DELETE**
    - Se la tupla dell'impiegato che supervisiona viene cancellata, il valore di SUPERSSN è posto a null in tutte le tuple impiegato che lo referenziano.
  - **CASCADE ON UPDATE**
    - Se il valore SSN di un impiegato che supervisiona è aggiornato, il nuovo valore è riportato in SUPERSSN di tutte le tuple impiegato che referenziano il valore aggiornato.
- Ai vincoli può essere dato un nome (per poterli riutilizzare), usando la keyword **COSTRAINT**.



# CHECK CONSTRAINT

- Il vincolo CHECK serve a controllare che un determinato attributo rispetta determinate condizioni.
- La sintassi è **CHECK(*vincolo*)**

dove vincolo è una determinata condizione che deve essere rispettata

```
CREATE DOMAIN sesso AS CHAR  
    CHECK(sesso = 'm' OR sesso='f')
```

```
CREATE DOMAIN voto AS SMALLINT  
    CHECK(voto> 0 AND voto<=30)
```

```
CONSTRAINT checkLode  
    CHECK(lode=TRUE AND voro=30) OR NOT lode=TRUE
```



```
CREATE Table impiegato(
    CF codice_fiscale,
    fname VARCHAR(15),
    mname CHAR,
    lname VARCHAR(15),
    gender CHAR,
    dno int,
    salary DECIMAL(10,2),
    CONSTRAINT emppk PRIMARY KEY(CF),
    CONSTRAINT fnameNN CHECK(fname IS NOT NULL),
    CONSTRAINT lnameNN CHECK(lname IS NOT NULL),
    CONSTRAINT depFK FOREIGN KEY(dno) REFERENCES dipartimento(dnumber),
    CONSTRAINT genderCorrect CHECK(gender='m' OR gender='f'),
    CONSTRAINT salaryCheck CHECK(salary>1000.00),
    CONSTRAINT cfnumberUNIQUE UNIQUE(CF, dno)
)
```



# RECAP VINCOLI

- CONSTRAINT <nome\_vincolo> <vincolo>
- <vincolo>:= CHECK <espressione booleana>
  - UNIQUE (<lista\_attributi>)
  - PRIMARY KEY (<lista attributi>)
  - FOREIGN KEY (<lista\_attributi\_FK>) REFERENCES <nome\_tabella>(<lista\_att\_PK>)
    - [ON DELETE <azione>]
    - [ON UPDATE <azione>]
- <azione>:= NO ACTION | CASCADE | SET DEFAULT | SET NULL



# DAL DDL ALL'SQL

METADATI

Definizione

CREATE TABLE

Modifica

ALTER TABLE

Cancellazione

DROP TABLE

INSERT

UPDATE

DELETE

DATI

SELECT

Interrogazione





# FINE

Per eventuali domande: (in ordine di preferenza personale)

- Ora.
- Chat di Teams
- Mail: [silvio.barra@unina.it](mailto:silvio.barra@unina.it)

