Corso di fondamenti di informatica – prima parte

Ingegneria industriale - polo di Frosinone

Indice

- **Fondamenti concettuali di informatica……pag. 2**
- **Architettura di un elaboratore ……pag. 14**
- **Hardware……pag. 21**
- **Software di sistema e applicativo……pag. 36**
- **Informatica e società……pag. 39**
- **Modello dei dati e database……pag. 50**
- **Il linguaggio SQL……pag. 102**
- **Microsoft Access……pag. 138**

"La forza degli individui risiede nella loro capacità di apprendere, cioè di dominare l'informazione, di assimilarla, di trasformarla in conoscenza e di utilizzarla in modo rapido ed efficace."

(F. Henri & C. R. Rigault - 1996).

Fondamenti concettuali di informatica

L'informatica è una scienza interdisciplinare, poiché impiegata in tutte le altre scienze ed interessa moltissimi aspetti della vita di tutti i giorni. La parola deriva dall'unione dei termini «*informazione*» e «*automatica*», in quanto questa scienza si occupa della trasformazione (o elaborazione) di informazioni in modo automatico cioè senza il diretto e continuo intervento dell'uomo.

L'informazione è costituita da un insieme di dati elementari, di diversa provenienza e natura, associati alle regole e alle istruzioni che permettono al microprocessore di elaborarli.

Una sequenza logicamente ordinata di operazioni o di istruzioni costituisce un **programma**: l'esecuzione, da parte di un elaboratore, di questo insieme di istruzioni produce la soluzione a un problema dato.

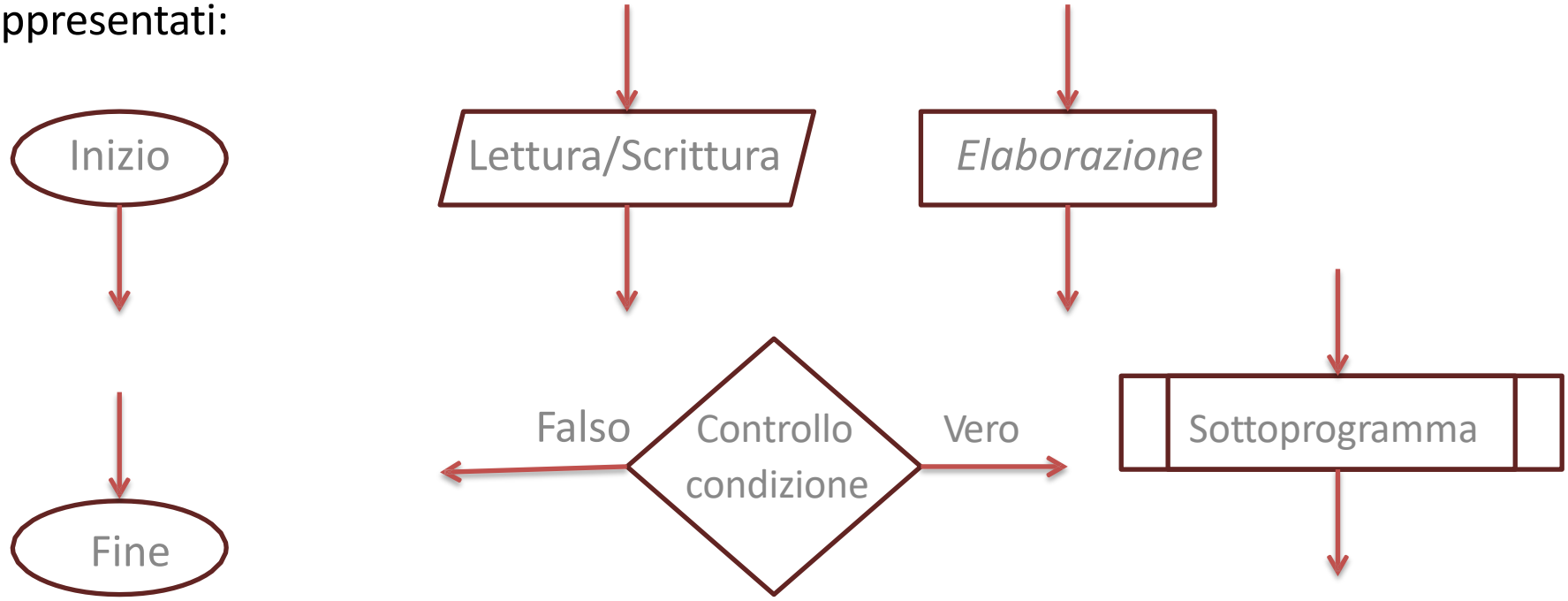
In informatica un programma è la descrizione di un algoritmo che permette la risoluzione, in un numero finito di passi, di una determinata classe di problemi.

L'algoritmo è la descrizione di un insieme finito di passi o istruzioni che devono essere eseguite per portare a termine un dato compito e per raggiungere un risultato.

Fondamenti concettuali di informatica

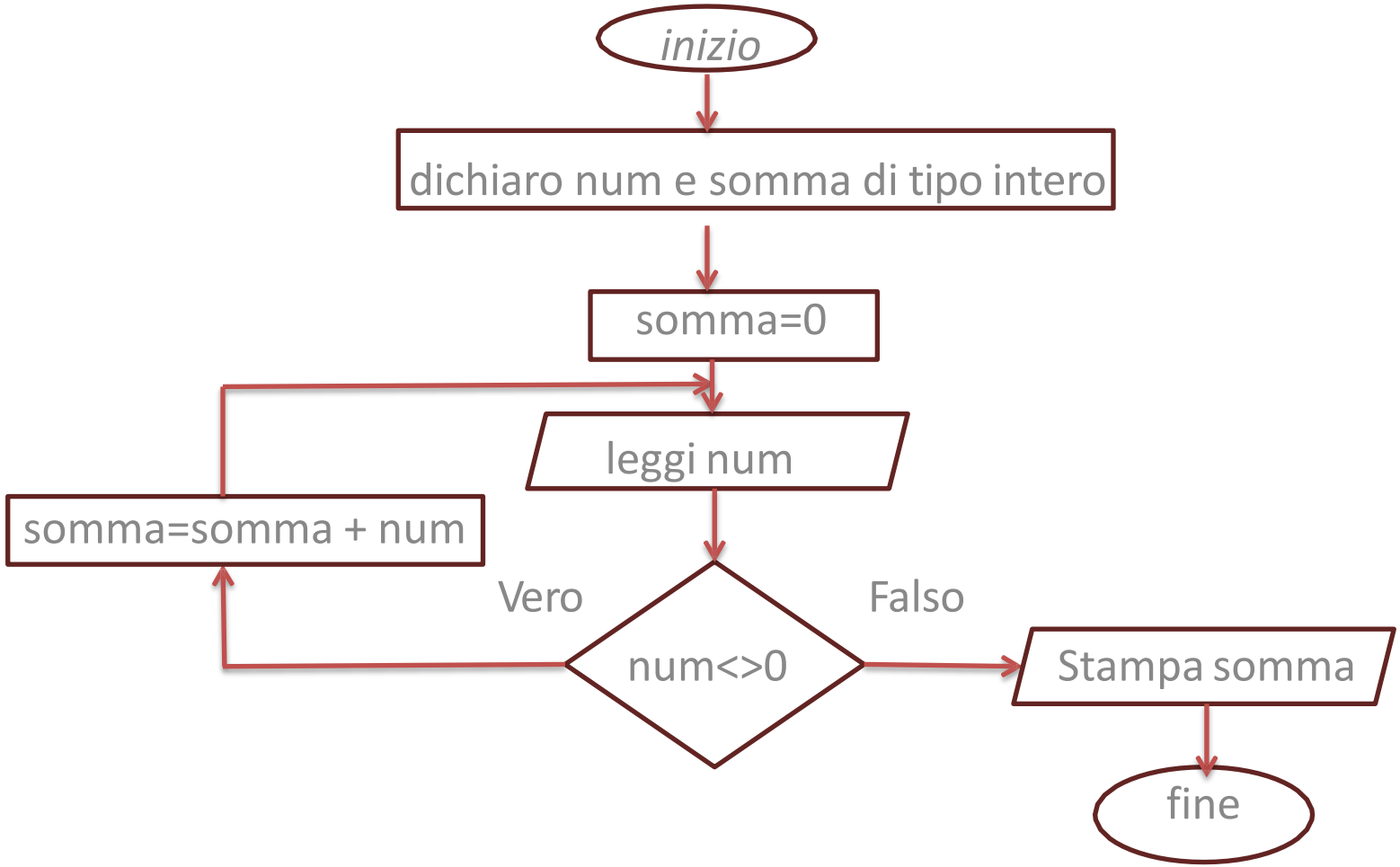
Rappresentare gli algoritmi con il diagramma di flusso.

Possiamo dare una rappresentazione grafica della risoluzione di un problema: gli algoritmi, infatti, si possono rappresentare come diagrammi di flusso o flow chart. Un flow chart ci permette di descrivere il processo in tutte le sue parti: ogni fase e ogni condizione del processo sono rappresentate da simboli grafici, detti «blocchi elementari», collegati tra loro tramite frecce che indicano la sequenza temporale nella quale avvengono. I blocchi elementari sono di sei tipi diversi e sono così rappresentati:



Rappresentare gli algoritmi con il diagramma di flusso.

Esercizio n. 1: Sommare i numeri inseriti da tastiera e stampare la somma ottenuta, per terminare l'input inserire uno zero.



Sistemi di numerazione

I sistemi di numerazione servono a rappresentare i numeri grazie a un insieme finito di simboli elementari, detti cifre.

Nel **sistema decimale** i numeri, quindi, sono creati utilizzando un insieme di dieci cifre, comprese tra 0 e 9, che assumono pesi diversi a seconda della posizione che occupano. Questi pesi sono sempre esprimibili come potenze di 10.

Se prendiamo il numero 2.089 in base dieci, possiamo considerarlo come il risultato della seguente addizione:

$2.089=9x10^0+8x10^1+0x10^2+2x10^3=9+80+0+2000$ ➔ 2.089

Gli elaboratori elettronici, però, non possono usare il sistema numerico decimale: sono realizzati, infatti, secondo le tecniche che appartengono all'elettronica digitale, nella quale è molto semplice costruire oggetti che assumono solo uno tra due stati stabili. Si consideri che il passaggio tra uno stato e l'altro dell'elemento, detto anche **commutazione**, è molto veloce. Il sistema di numerazione usato dagli elaboratori è il **sistema binario**: la sua base è infatti il numero 2 e i simboli che operano sono 0 e 1 . Nel sistema binario, quindi, tutti i numeri saranno espressi sulla base delle potenze di 2.

Se prendiamo il numero 1010 in base due, possiamo considerarlo come il risultato della seguente addizione:

$1010=0X2^0+1x2^1+0x2^2+1x2^3=0+2+0+8$=> 10 in base decimale

Per convertire un numero binario in decimale, quindi, dobbiamo moltiplicare ogni cifra, a partire da destra, per 2 elevato alla sua posizione, iniziando da zero, e quindi sommare i risultati ottenuti.

Conversione decimale - binario

Per convertire un numero intero decimale X in un numero binario dobbiamo eseguire una serie di divisioni ripetute: si divide il numero per 2 e si ottiene un resto e un quoziente intero; a questo punto si divide il quoziente ottenuto per 2, e si prosegue così fino ad ottenere un quoziente pari a 0. Il numero binario risultante è dato da tutti i resti ottenuti a partire dall'ultimo.

Esempio:

73 (decimale)=? (binario)

73/2=36 resto 1

36/2= 18 resto 0

18/2=9 resto 0

9/2=4 resto 1

4/2=2 resto 0

2/2=1 resto 0

1/2=0 resto 1

73(decimale)=1001001(binario)

Le cifre del sistema binario sono dette bit, contrazione di **binary digit**, in italiano cifra binaria. Otto bit formano un **byte** (unità elementare dell'informazione).

Rappresentazione dei caratteri e nozione di bit e byte

Per rappresentare informazioni diverse all'interno del computer si utilizzano pacchetti di bit: se un bit, infatti, ci offre due possibilità (0 e 1), due bit ce ne offrono quattro (00, 01, 10, 11) , tre bit offrono otto possibilità di rappresentazione dell'informazione e così via. Un pacchetto do otto bit ci mette a disposizione ben 256 possibilità, cioè definisce 256 sequenze differenti di bit. Questo pacchetto di otto bit prende il nome di byte ed è, in pratica, l'unità minima di memorizzazione dell'elaboratore. Sulla base di queste considerazioni, nel 1961 un ingegnere dell'IBM, Bob Bemer, propose un sistema di codifica dei caratteri, assegnando a ognuno di essi una sequenza di bit, cioè un byte. Questa codifica, che prese il nome di ASCII (America Standard Code for Information Interchange, ovvero Codice Standard Americano per lo Scambio di Informazioni), è quello attualmente più utilizzato negli elaboratori. In essa, ad esempio, la lettera «A» corrisponde al byte 65 e, quindi, ogni volta che scriviamo una «A» con la tastiera, l'elaboratore, in realtà, processa la sequenza di bit 01000001. Nel codice ASCII troviamo tutte le lettere maiuscole e minuscole, tutte le cifre, tutte le lettere accentate e i segni di punteggiatura, caratteri diversi, comandi (ad esempio il tasto di invio corrisponde al codice ASCII 13, cioè alla sequenza di bit 00001101).

Codice ASCII

0		32		64	@	96	`	128	Ç	160	á	192	└	224	Ó
1	☺	33	!	65	A	97	a	129	ü	161	í	193	┴	225	ß
2	☻	34	"	66	B	98	b	130	é	162	ó	194	┬	226	Ô
3	♥	35	#	67	C	99	c	131	â	163	ú	195	├	227	Ò
4	♦	36	$	68	D	100	d	132	ä	164	ñ	196	─	228	õ
5	♣	37	%	69	E	101	e	133	à	165	Ñ	197	┼	229	Õ
6	♠	38	&	70	F	102	f	134	å	166	ª	198	ã	230	µ
7	•	39	'	71	G	103	g	135	ç	167	º	199	Ã	231	þ
8	◘	40	(	72	H	104	h	136	ê	168	¿	200	╚	232	Þ
9	○	41	)	73	I	105	i	137	ë	169	®	201	╔	233	Ú
10	◙	42	*	74	J	106	j	138	è	170	¬	202	╩	234	Û
11	♂	43	+	75	K	107	k	139	ï	171	½	203	╦	235	Ù
12	♀	44	,	76	L	108	l	140	î	172	¼	204	╠	236	ý
13	♪	45	-	77	M	109	m	141	ì	173	¡	205	═	237	Ý
14	♫	46	.	78	N	110	n	142	Ä	174	«	206	╬	238	¯
15	☼	47	/	79	O	111	o	143	Å	175	»	207	¤	239	´
16	►	48	0	80	P	112	p	144	É	176	░	208	ð	240	-
17	◄	49	1	81	Q	113	q	145	æ	177	▒	209	Ð	241	±
18	↕	50	2	82	R	114	r	146	Æ	178	▓	210	Ê	242	‗
19	‼	51	3	83	S	115	s	147	ô	179	│	211	Ë	243	¾
20	¶	52	4	84	T	116	t	148	ö	180	┤	212	È	244	¶
21	§	53	5	85	U	117	u	149	ò	181	Á	213	ı	245	§
22	▬	54	6	86	V	118	v	150	û	182	Â	214	Í	246	÷
23	↨	55	7	87	W	119	w	151	ù	183	À	215	Î	247	¸
24	↑	56	8	88	X	120	x	152	ÿ	184	©	216	Ï	248	°
25	↓	57	9	89	Y	121	y	153	Ö	185	╣	217	┘	249	¨
26	→	58	:	90	Z	122	z	154	Ü	186	║	218	┌	250	·
27	←	59	;	91	[	123	{	155	ø	187	╗	219	█	251	¹
28	∟	60	<	92	\	124	|	156	£	188	╝	220	▄	252	³
29	↔	61	=	93	]	125	}	157	Ø	189	¢	221	¦	253	²
30	▲	62	>	94	^	126	~	158	×	190	¥	222	Ì	254	■
31	▼	63	?	95	_	127	⌂	159	ƒ	191	┐	223	▀	255	

Dal sistema binario all'algebra di Boole

Il sistema binario, creato nel secolo 1600 da parte di ***Gottfried Leibnitz matematico, filosofo, scienziato, logico, glottoteta, diplomatico, giurista, storico, magistrato tedesco di origine sorba*** (serbi di Lusazia), è un sistema con due simboli.

Quindi, se per bit intendiamo un simbolo che può assumere solo due valori, spesso indicati sulla carta 0/1, Vero/Falso, On/Off, a livello circuitale è rappresentato per esempio da livello di tensione 0 Volt oppure 5 Volt. Interponendo 0 e 1 come cifre di un sistema di numerazione di base 2, si possono definire alcune semplici **operazioni sia aritmetiche sia logiche**.

Ritratto di Gottfried Wilhelm von Leibniz conservato presso la Biblioteca regionale di Hannover.

Tali operazioni sono stata definite da George Boole nel 1854 per la manipolazione di espressioni logiche attraverso modelli matematici , l'algebra di Boole opera su variabili che possono assumere solamente due valori: 0 e 1 (variabili logiche booleane) .

Con queste variabili si possono rappresentare eventi tipicamente binari: affermazioni che possono essere VERE oppure FALSE. Utilizzando le variabili booleane, si costruiscono le funzioni booleane o logiche, definite attraverso opportune tavole della verità.

L'algebra di Boole


George Boole

Una funzione booleana ha una o più variabili in input e fornisce risultati che dipendono solo da queste variabili.

Poiché le variabili possono assumere solo i valori 0 o 1 una funzione booleana con n variabili di input ha solo 2^n combinazioni possibili e può essere descritta dando una tabella, detta tabella di verità, con 2^n righe.

Le variabili si indicano con le lettere A,B,C,X,Y,W,Z. Le operazioni base sono AND (•), OR (+) ,NOT (¯)

X=f(A,B)

OR (+)

A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

AND(•)

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

NOT (¯)

A	X
0	1
1	0

Le porte logiche

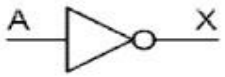
Sono stati individuati e costruiti circuiti elettronici che realizzano le operazioni elementari, questi sono detti PORTE LOGICHE.

Le porte logiche sono circuiti che operano su più segnali di ingresso e producono un segnale di uscita.

Rispondono a due valori di range di tensioni che associamo ai valori logico 0 e 1.

Le porte logiche che rappresentano le operazioni AND, OR, NOT, NAND, NOR e XOR sono di seguito riportate.

NOT

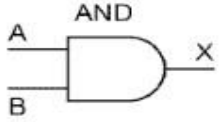


A	X
0	1
1	0

OR

A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

AND

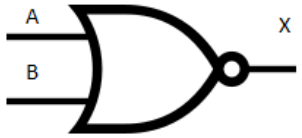


A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

NAND

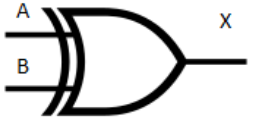
A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

NOR



A	B	X
0	0	1
0	1	0
1	0	0
1	1	0

XOR



A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

Il Codice BCD

Il codice BCD (Binary Coded Decimal), permette la codifica binaria del codice decimale mediante quattro bit. In questo formato ogni cifra di un numero è rappresentata da un codice binario di quattro bit , il cui valore è compreso tra 0 (0000) e 9 (1001). Le restanti sei combinazioni possono essere usate per rappresentare simboli. Per esempio il numero 127 è rappresentato in BCD come 0001- 0010- 0111.

Se vogliamo rappresentare allora un numero decimale in codifica BCD, abbiamo ad esempio:

Numero decimale da convertire $756_{(10)}$

7=0111 5=0101 6=0110

Numero in notazione BCD : $0111\text{-}0101\text{-}0110_{(BCD)}$

Somma in BCD - Per effettuare la somma di due numeri BCD si può usare la classica tecnica binaria.

Ad esempio :

PRIMO ADDENDO	156	0 0 0 1 | 0 1 0 1 | 0 1 1 0
SECONDO ADDENDO	423	0 1 0 0 | 0 0 1 0 | 0 0 1 1
RESTI		0 0 0 0 0 | 0 0 0 0 | 1 1 0
RISULTATO		0 1 0 1 | 0 1 1 1 | 1 0 0 1

Codice BCD

Può accadere, però, che la **somma** di due cifre possa dare un risultato maggiore di 9, per cui il nibble (4 bit) corrispondente non sarebbe un codice BCD corretto come nel seguente esempio

PRIMO ADDENDO	156	0 0 0 1 | 0 1 0 1 | 0 1 1 0
SECONDO ADDENDO	454	0 1 0 0 | 0 1 0 1 | 0 1 0 0
RESTI		0 0 0 0 0 1 0 1 0 1 0 0
RISULTATO		0 1 0 1 | 1 0 1 0 | 1 0 1 0

In questo caso si nota come il secondo e terzo nibble non corrispondano ad un codice BCD valido. Si può correggere il risultato sommando la stringa 0110 ai nibble non corretti .

PRIMO ADDENDO	156	0 0 0 1 | 0 1 0 1 | 0 1 1 0
SECONDO ADDENDO	454	0 1 0 0 | 0 1 0 1 | 0 1 0 0
RESTI		0 0 0 0 0 1 0 1 0 1 0 0
RISULTATO		0 1 0 1 | 1 0 1 0 | 1 0 1 0
FATTORE CORRETTIVO		0 0 0 0 | 0 1 1 0 | 0 1 1 0
RESTI		0 0 0 1 1 1 1 0 1 1 1 0
		0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0

risultato: $0110\text{-}0001\text{-}0000_{(BCD)} = 610_{(10)}$

Veri sistemi di numerazione

Binary Base-2	Decimal Base-10	Hexa-Decimal Base-16	Octal Base-8	BCD Code	Gray Code
0000	0	0	0	0	0000
0001	1	1	1	1	0001
0010	2	2	2	2	0011
0011	3	3	3	3	0010
0100	4	4	4	4	0110
0101	5	5	5	5	0111
0110	6	6	6	6	0101
0111	7	7	7	7	0100
1000	8	8	10	8	1100
1001	9	9	11	9	1101
1010	10	A	12	---	1111
1011	11	B	13	---	1110
1100	12	C	14	---	1010
1101	13	D	15	---	1011
1110	14	E	16	---	1001
1111	15	F	17	---	1000

Trasformare il numero 20 decimale in numero ottale:

20:8 = 2 resto 4

2:8 = 0 resto 2 il risultato è: 24 (8)

Convertire il binario in Gray

Per convertire un numero binario in codice Gray è molto semplice:

Si effettua un'operazione di XOR tra il numero binario e se stesso shiftato di una posizione a destra. Riportiamo di seguito un esempio:

Binario: 0110

Xor: 0110

Gray 0101

Da notare che la prima cifra del codice Gray è sempre la stessa del numero binario.

Utilizzo del codice Gray.

Molti dispositivi elettronici indicano la propria posizione chiudendo e aprendo degli interruttori. Se queste posizioni venissero rappresentate come quelle di una codifica binaria ordinata in modo naturale, il passaggio ad esempio dalla posizione 3 alla 4, anche se consecutivi, avrebbe tutti i bit di valore diverso (vedi tabella a sinistra). Questa situazione è difficile da rappresentare con degli interruttori fisici reali: è molto improbabile che tutti gli interruttori cambino il proprio stato (aperto/chiuso) nello stesso istante di tempo, creando così degli errori.

Con un sistema Gray il passaggio da una combinazione a quella successiva, vede modificare solo un bit per volta.

Architettura di un elaboratore elettronico

Che cos'è un elaboratore elettronico?

Con il termine elaboratore elettronico o computer intendiamo una macchina costituita da parti elettriche, meccaniche, ottiche, ecc, in grado di elaborare dati in modo automatico, veloce, sicuro ed efficiente.

Il termine **architettura riferito ad un elaboratore elettronico** indica l'organizzazione logica delle componenti interne di un computer e le modalità secondo le quali queste cooperano tra di loro per compiere azioni più o meno complesse.

Negli anni Cinquanta **John Von Neumann** ideò un computer, molto semplice se paragonato agli attuali ma decisamente innovativo all'epoca, chiamata appunto Macchina di Von Neumann. Si tratta di una macchina dotata di un esecutore (Central Processing Unit) in grado di scandire un elenco di istruzioni (programmi) e accedere a dati memorizzati in un organo di memoria (memoria centrale); l'esecutore è in grado anche di scambiare dati con delle locazioni di memoria presenti su particolari dispositivi, dette interfacce di I/O, che provvedono ai trasferimenti con i dispositivi periferici che consentono lo scambio di informazioni con l'esterno, quali monitor e stampanti .

Architettura di un elaboratore elettronico

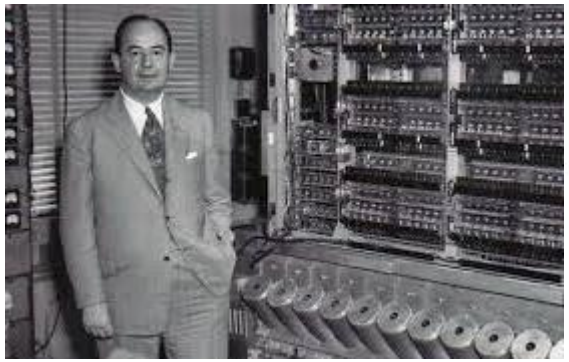
In relazione alla specifica realizzazione della CPU, è definito un insieme di sequenze binarie, normalmente di lunghezza multipla di 8 bit, dette istruzioni macchina, che specificano precise azioni da eseguire. CPU, MC, interfacce di I/O sono collegate attraverso un sistema strutturato di conduttori detto bus.

Si hanno tre tipi di bus:

-bus dati

-bus indirizzo

-bus controllo



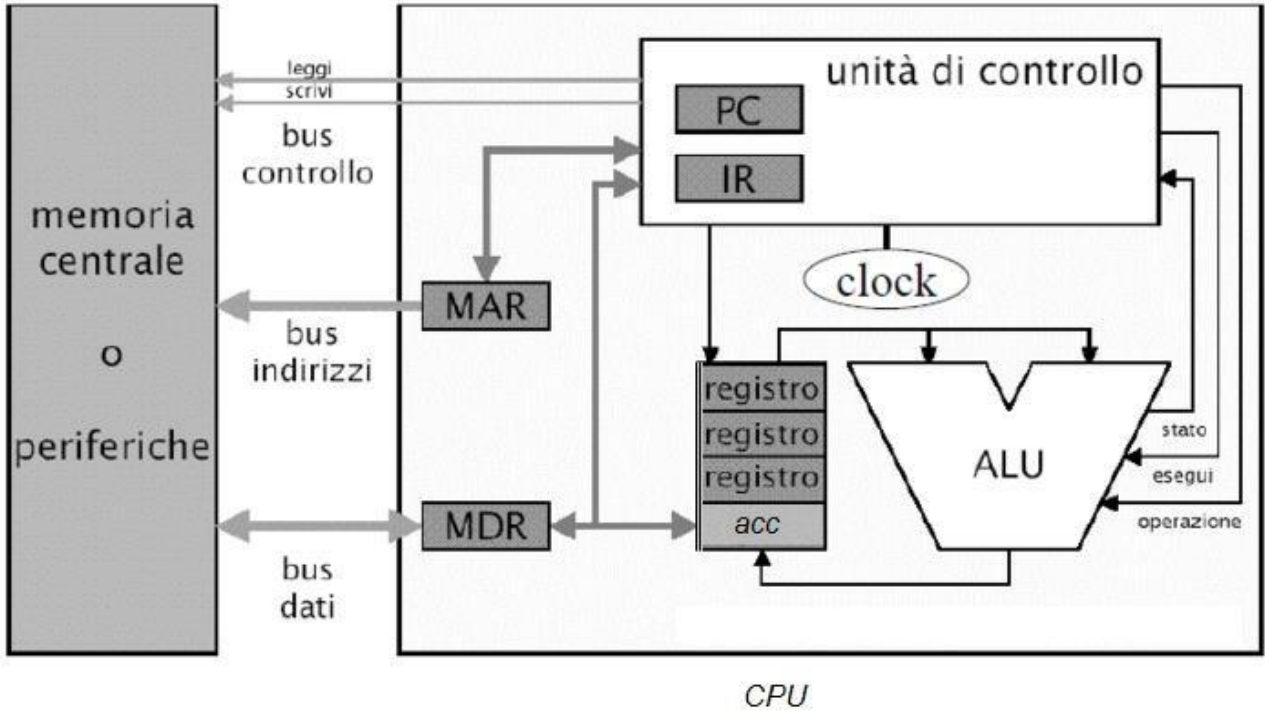


Foto e schema del modello di Von Neumann

Architettura di un elaboratore elettronico

La **CPU** (Central Processing Unit) o comunemente chiamato processore è il vero cervello del computer . Fondamentalmente la CPU è composta da due elementi fondamentali: l'ALU (unità logico-aritmetica), dove avvengono tutti i passaggi dell'elaborazione e la CU (control unit o unità di controllo), che stabilisce la logica con cui devono essere effettuate le singole operazioni dall'unità logico-artimetica.

La velocità del processore, infine, è data dal numero di cicli di istruzione al secondo che esso compie. Per comprendere il concetto di ciclo di istruzione, possiamo dire che in ciascun ciclo il processore:

- preleva l'istruzione;
- la interpreta;
- la esegue;
- passa all'istruzione successiva.

La velocità con cui può essere compiuto un ciclo di istruzione è determinata dal clock, che possiamo immaginare come un orologio interno che scandisce i tempi del processore: ad esempio, un processore che funziona a 1.800 Mhz (Mega Hertz), può compiere un miliardo ed 800 milioni di cicli di istruzioni al secondo. La CPU ha il compito di eseguire il ciclo istruzione, in pratica, intesa come black box, è in grado di effettuare operazioni di lettura/scrittura della memoria centrale o dei registri di I/0 e di interpretare i codici delle istruzioni generando tutti i segnali elettrici necessari a svolgere l'operazione richiesta.

Architettura di un elaboratore elettronico

ALU (unità logico-aritmetica). All'ALU è demandato il compito di effettuare le operazioni logiche e aritmetiche sulle sequenze binarie lette dalla memoria centrale o dai registri di I/O. E' importante capire che l'ALU è un sistema combinatorio ed è caratterizzata da quanti bit può elaborare simultaneamente: si definisce così il grado di parallelismo della CPU.

UC (Unità di controllo). E' l'unità di governo dell'intera CPU; riceve in ingresso dal registro IR, che in seguito assumeremo a 8 bit, il codice dell'istruzione macchina; in base a esso vengono generati tutti i segnali di temporizzazione dei dispositivi interni ed esterni. Si tratta di un sistema sequenziale le cui transizioni sono cadenzate dal clock.

IR (Instruction Register). Il registro istruzioni memorizza il codice operativo dell'istruzione macchina in fase di esecuzione. Si parla di codice operativo perché l'istruzione macchina potrebbe contenere anche informazioni circa gli operandi.

PC (Program Counter). Contatore di programma, è un registro che contiene l'indirizzo della prossima istruzione da eseguire. E' di fondamentale importanza in quanto contiene sempre l'indirizzo della prossima istruzione da eseguire rispetto a quella attuale. Se il program counter ha 16 bit, allora si possono indirizzare 65536 possibili locazioni diverse di memoria centrale (**2^{16}** locazioni di memoria)

ACC (Registro accumulatore). Il registro contiene sempre il risultato delle operazioni aritmetiche ed è sempre coinvolto nelle operazioni di ALU. Fornisce uno (o l'unico) operando e memorizza il risultato.

MDR (Buffer dati). Si tratto di un registro non manipolabile a livello istruzione macchina, ma necessario per collegare i circuiti interni della CPU al bus, parte dati.

MAR (Buffer Indirizzi). Ciò che avviene per il buffer dati si ripete per il buffer indirizzi, con l'unica differenza che quest'ultimo è collegato al bus indirizzi.

Registri generali. I registri generali dati memorizzano temporaneamente i dati nella CPU, eventualmente inviandoli alla ALU come secondo operando.

Il registro PWS

PSW (Program Status Word) o registro di stato. Oltre al risultato vero e proprio di una operazione logica aritmetica, si registrano alcuni stati binari (flag di condizione) di fine operazione.

Flag di condizione più importanti:

- **ZF** = Zero flag (o flag zero). Viene posto a 1 dalla CPU se il risultato di un'operazione matematica o logica è zero.
- **CF** = Carry flag (o flag di riporto). Viene posto a 1 dalla CPU se il risultato di un'operazione produce un riporto.
- **SF** = Sign flag (flag di segno). Viene posto a 1 dalla CPU se il risultato dell'ultima operazione ha segno negativo.
- **OF** = Overflow flag. Viene posto a 1 dalla CPU quando il risultato dell'ultima operazione è maggiore del massimo valore che il registro di destinazione può contenere. È simile al carry flag, ma viene impiegato nelle operazioni in cui è presente il segno degli operandi.
- **AF** = Auxiliary Carry Flag. usato nell'aritmetica BCD; vale 1 se c'è stato riporto (somma) o prestito (sottrazione) del bit 3
- **PF** = Parity Flag. Viene posto a 1 dalla CPU se l'ultima operazione ha dato come risultato un numero binario pari, in senso booleano (cioè l'ultimo bit è 0)

I BUS di interconnessione

BUS

Il bus è il sistema di interconnessione strutturato che fornisce i percorsi per il trasferimento dei dati, degli indirizzi e dei segnali di controllo necessari allo scambio dei dati tra gli altri moduli. Fisicamente consiste in piste tracciate su circuito stampato e alcuni componenti elettronici per mantenere la correttezza delle forme d'onda dei segnali trasmessi.

Bus dati: è un bus bidirezionale e condiviso da tutti i moduli, quindi sono previste regole per evitare conflitti. Tramite i segnali di controllo si decide chi debba assumere la gestione del bus. Sul bus dati viaggiano i dati e vanno da un dispositivo all'altro.

Bus Indirizzi: il bus indirizzi è unidirezionale ed è utilizzato per selezione una locazione di memoria o un'unità di input/output. Se il bus indirizzi è a n bit, si possono indirizzare 2^n locazioni di memoria centrale.

Bus controlli: ha diversi segnali, tra cui certamente gli impulsi (strobe) di comando della lettura o della scrittura in memoria o su un dispositivo di I/O, i segnali di interruzione che consentono a un'interfaccia di I/O di chiedere servizio alla CPU, ed altri controlli.

L’hardware

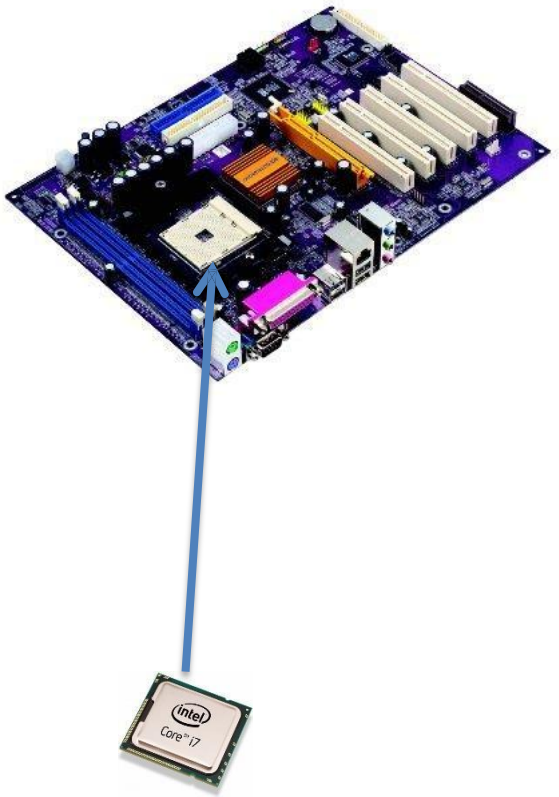
La struttura fisica di un computer, chiamata nel suo insieme hardware, comprende:

- un gruppo di componenti elettronici che costituiscono il nucleo centrale dell’elaboratore, sistemati all’interno di un involucro metallico chiamato **case** (scatola);
- alcune apparecchiature che consentono l’ingresso delle informazioni (periferiche di input);
- alcune apparecchiature che consentono l’uscita delle informazioni (periferiche di output);
- diversi attacchi, dette interfacce o porte, per il collegamento tra il nucleo centrale e le periferiche.

Facendo riferimento a un normale PC da scrivania, descriveremo brevemente questi dispositivi, iniziando dai componenti interni al **case**.

La scheda madre

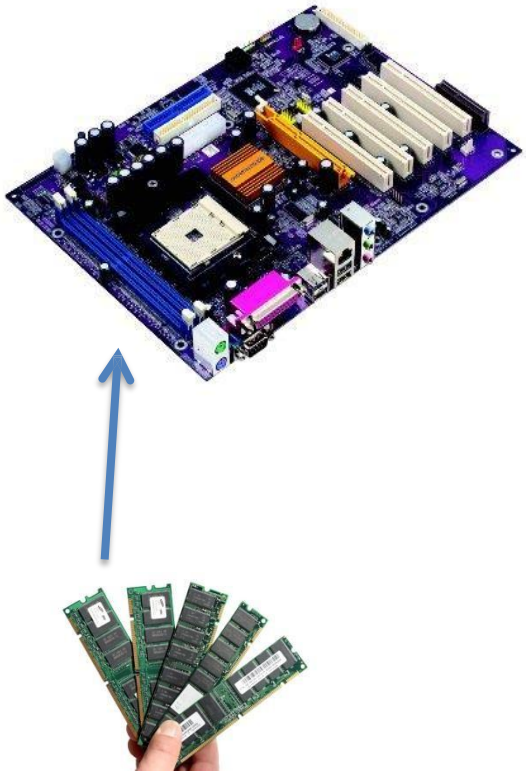
All'interno del case troviamo innanzi tutto la cosiddetta scheda madre, una grossa piastra contenente moltissimi dispositivi elettronici, alla quale sono collegati tutti i componenti del computer. Essa, grazie a una serie di chip, dirige il traffico delle informazioni, indirizzandole e distribuendole adeguatamente tra i diversi elementi collegati. La scheda madre rappresenta il cuore del PC: le sue caratteristiche determinano la potenza di elaborazione, la possibilità di aggiornamenti futuri, l'assortimento delle periferiche collegabili.



La CPU (unità centrale di elaborazione)

L"unità centrale di elaborazione, già analizzata in precedenza, è realizzata mediante un unico chip, chiamato microprocessore, alloggiato nella scheda madre. La CPU costituisce il cervello del sistema, in quanto controlla tutte le funzioni del PC ed è capace di eseguire in sequenze tutte le operazioni elementari che gli vengono richieste da una serie di istruzioni (programma).

La memoria RAM

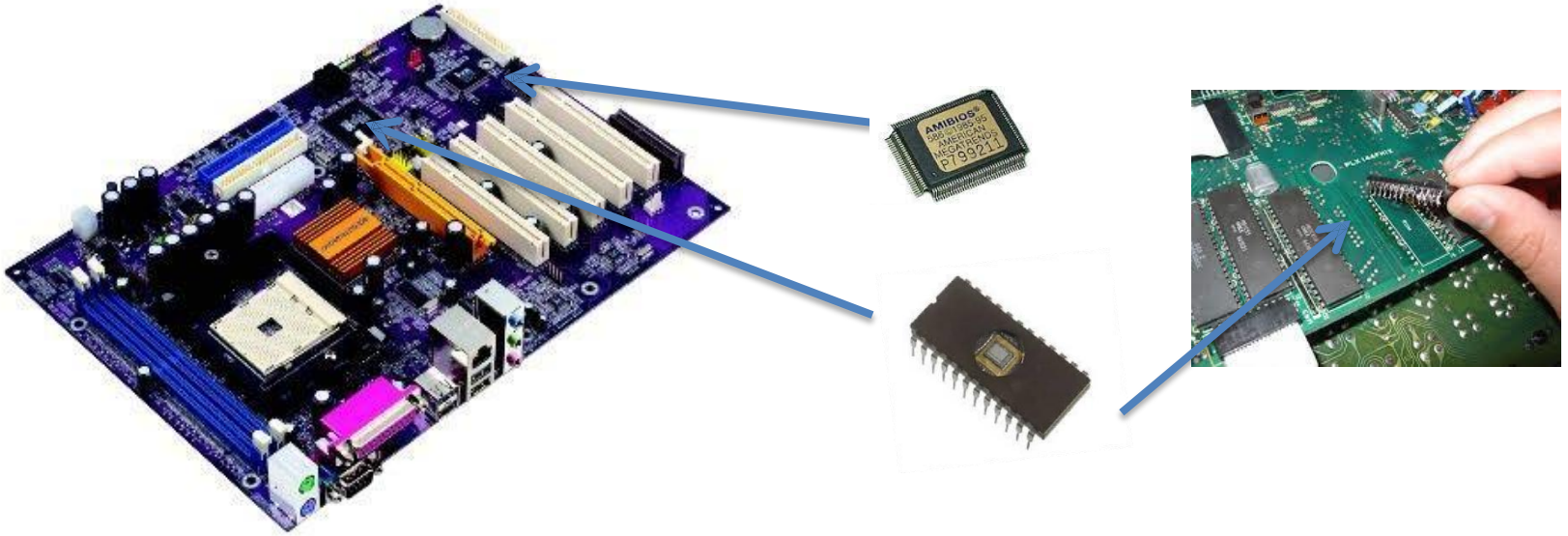


La sigla RAM (Random Access Memory: memoria ad accesso casuale) indica la memoria interna principale del PC, costituita fisicamente da alcuni moduli alloggiati sulla scheda madre. Essi sono suddivisi in celle affiancate, ciascuna delle quali è identificata da due numeri (indirizzo di memoria). La RAM contiene i dati su cui lavorare e i programmi da eseguire. Essa comunica da una parte con la CPU, che legge le istruzioni e poi rinvia i risultati delle elaborazioni, e dall'altra con memorie esterne e unità periferiche. Il processore può accedere a ciascuna cella della memoria RAM in qualsiasi ordine (da cui il termine «casuale»). La memoria RAM è di tipo volatile, ossia allo spegnimento del PC se ne perde il contenuto.

Le sue caratteristiche essenziali sono la **capacità**, che si misura in gigabyte (un gigabyte è pari a circa un miliardo di byte, ossia 10^9 byte), e il **tempo di accesso,** cioè il tempo che la RAM impiega per fornire alla CPU i dati richiesti. Poiché gli accessi alla RAM risultano più lenti rispetto alla velocità degli attuali processori, i dati impiegati maggiormente vengono duplicati in un sottoinsieme della memoria (cache memory, molto veloce e di piccola capacità). Quando il processore fa riferimento a un indirizzo RAM, la cache controlla se lo contiene e, in caso affermativo, fornisce i dati al processore.

La memoria ROM

La ROM (Read Only Memory: memoria di sola lettura) contiene istruzioni inserite dalla Casa costruttrice del PC che possono essere eseguite ma non modificate. È una memoria di tipo permanente, perché il suo contenuto non si cancella allo spegnimento del PC. Nella ROM sono memorizzati i programmi fondamentali (BIOS: Basic Input/Output System) che gestiscono tutte le funzionalità di base di un PC. Essi si attivano automaticamente all'accensione del computer e, tra l'altro, gli consentono di riconoscere tutti i componenti del sistema e diagnosticare il loro corretto funzionamento. Sono sempre più diffuse, attualmente, anche memorie ROM che sono in realtà di sola lettura (EEPROM: Electronically Erasable Programmable ROM), perché possono essere riprogrammate per inserire aggiornamenti predisposti dal costruttore.



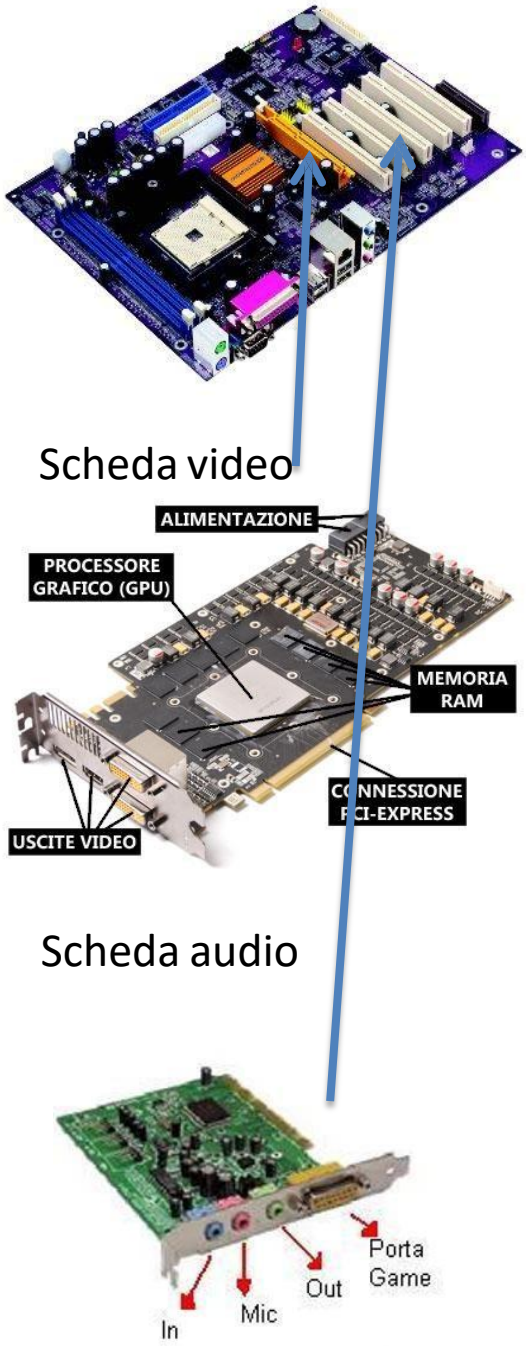
La scheda grafica

La scheda grafica consente di visualizzare su un monitor immagini e caratteri alfanumerici. Essa è dotata di una propria memoria RAM, destinata a contenere i dati da visualizzare, e di un proprio processore grafico che, attraverso un convertitore digitale-analogico, genera i segnali elettrici da inviare allo schermo.

Le attuali schede grafiche supportano anche la grafica 3D, mettendo a disposizione un processore grafico (GPU) dotato di speciali funzioni, in grado di velocizzare la costruzione e la visualizzazione di un'immagine con effetto tridimensionale

La scheda audio

Il PC multimediale elabora i suoni mediante una scheda audio montata sulla scheda madre nell'apposito (slot). Essa è dotata di un processore audio, che ha il compito di gestire tutti i calcoli relativi al suono, e di un convertitore digitale-analogico, collegato ad altoparlanti di uscita. È presente di norma anche un convertitore analogico-digitale, che consente di acquisire suoni dall'esterno attraverso un microfono. Di norma le attuali schede madri hanno a bordo già la propria scheda audio, ma volendo è possibile aggiungerne una esterna su slot PCI.



Le porte per periferiche

Le porte o interfacce per le periferiche sono situate generalmente nella parte posteriore del Case, dove si collegano i componenti esterni. Le porte sono di diverso tipo, ognuno dei quali è adatto per il collegamento di determinate periferiche. In generale, le porte si distinguono in porte seriali, sono piuttosto lente perché possono trasmettere un bit per volta, e in porte parallele, che consentono il trasferimento simultaneo di un byte (8 bit). Oltre alle porte specificatamente destinate a tastiera, mouse, monitor, citiamo soltanto la porta USB (Universal Serial Bus), che si è molto diffusa perché possiede caratteristiche notevoli. Si tratta infatti di un'interfaccia seriale più veloce di quelle tradizionali, che consente di collegare molte periferiche diverse anche a macchina accesa (connessione «a caldo»).

Posta PS2
(verde per mouse – viola per tastiera)

Poste USB

Scheda video integrata

Scheda di rete integrata

Posta per audio
(verde casse – blu porta in– rosa microfono)

Le memorie di massa : l'hard disk

Nei moderni computer la quantità di informazioni che occorre avere in memoria è molto alta, dell'ordine di miliardi di byte. Non è possibile contenere questa montagna di informazioni nella memoria centrale (RAM) , in quanto essa è costosa e allo stesso tempo volatile, pertanto non adatta alla conservazione permanente dei dati.

Per questi motivi vengono utilizzati altri dispositivi di memoria , le cosiddette memorie di massa o memorie ausiliarie.

I dispositivi di memoria di massa utilizzati su larga scala sono le **memorie magnetiche**, le **memorie ottiche** e le **memorie flash**.

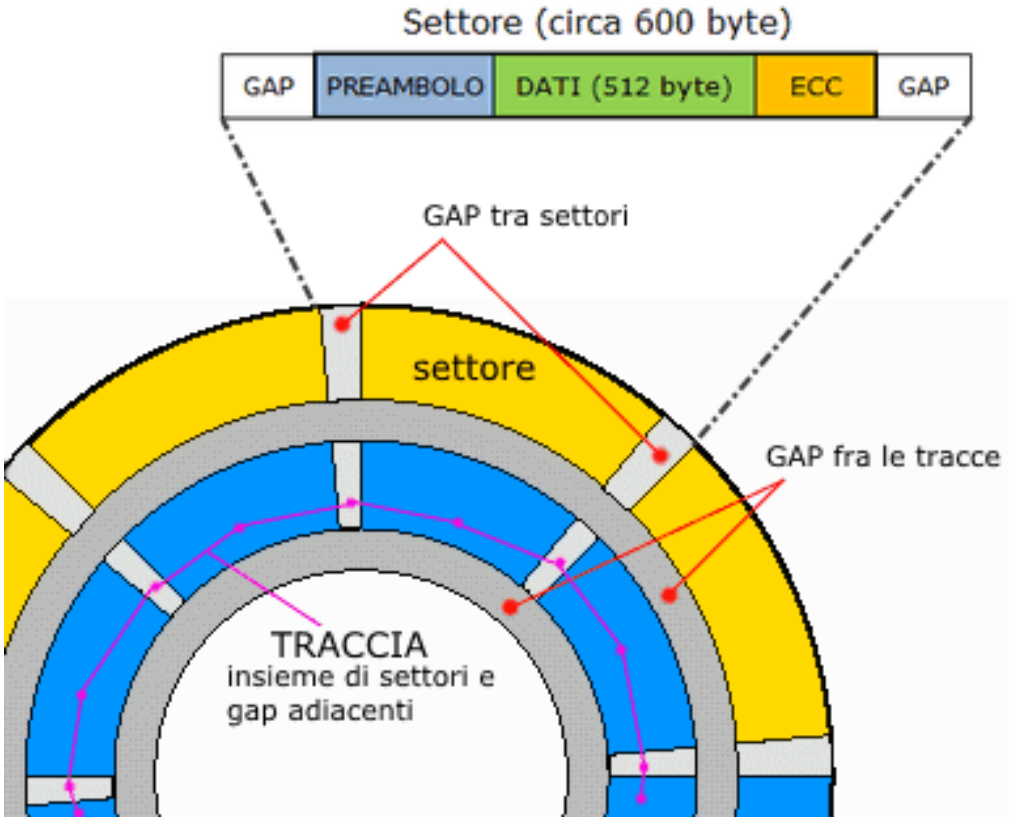
I dispositivi magnetici sono costituiti da un supporto piano ricoperto di materiale ferromagnetico sul quale è possibile memorizzare le informazioni magnetizzando apposite areole (l'equivalente delle celle di memoria). Questo fenomeno fisico ben si presenta alla memorizzazione di segnali digitali in quanto può assumere due stati in base alla direzione del campo magnetico. La registrazione e la lettura dei dati avvengono grazie a una testina di lettura/scrittura.

Tutte le memorie di massa magnetiche hanno in comune le seguenti caratteristiche:

- **Velocità di movimento** del supporto di memorizzazione, misurata in pollici/sec;
- **Densità di memorizzazione**, misurata in bit/pollice;
- **Velocità di trasferimento**, misurata in bit/sec e data dal prodotto dei due parametri precedenti.

Le memorie di massa : l'hard disk

Il disco magnetico è un supporto di memorizzazione di massa ad accesso diretto costituito da un piatto accuratamente levigato le cui facce sono ricoperte di materiale magnetico. La superficie del disco è composta da tantissime areole magnetizzabili nelle quali vengono memorizzati i dati binari tramite la testina di lettura/scrittura. Le areole si trovano su piste circolari concentriche denominate **tracce,** su cui i dati vengono memorizzati sequenzialmente, cioè uno di seguito all'altro. Ogni traccia è suddivisa in un numero fisso di **settori** , separati da zone neutre dette **gap.**



Le memorie di massa: chiavette USB

Chiamate anche **pendrive**, sono il sostituto dei floppy disk e hanno una capacità di memorizzazione elevata, possono raggiungere anche 1 Terabyte (TB) di memoria ed utilizzano tecnologie di trasferimento veloci (USB 3.0).

Una **chiave USB** (anche in inglese *USB flash drive*) è una memoria di massa portatile di dimensioni molto contenute (qualche centimetro in lunghezza e circa un centimetro di larghezza).

Nella chiave USB i dati sono memorizzati in una memoria flash, tipicamente di tipo NAND, contenuta al suo interno. La capacità è limitata unicamente dalla densità delle memorie flash impiegate, con il costo per megabyte che aumenta rapidamente per alte capacità.

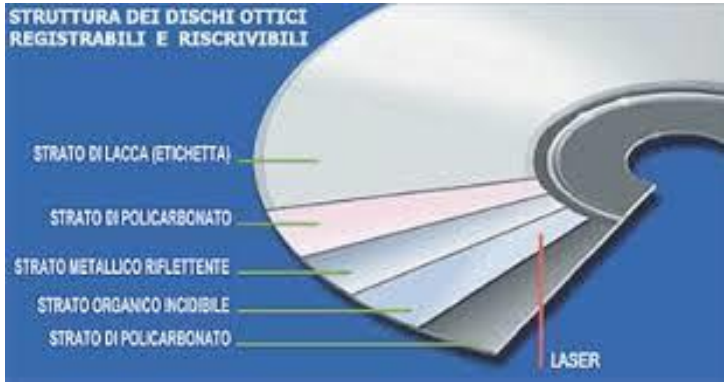
È da precisare però che la velocità non dipende solo dall'interfaccia, ma anche dal tipo di memoria flash utilizzata, e dalla eventuale presenza di microchip dedicati all'interno della chiavetta stessa. Esistono a questo proposito in commercio alcune chiavette che contengono un piccolo microprocessore dedicato ad ottimizzare il processo di lettura/scrittura sulla memoria flash. Grazie alle dimensioni ridotte, all'assenza di inaffidabili meccanismi mobili, alle crescenti dimensioni della memoria e alla sua interoperabilità, la chiavetta si sta configurando, accanto ai CD e ai DVD come unità preferita per il trasporto fisico di dati. Si tenga però in considerazione il fatto che il numero di scritture che una memoria flash può supportare non è illimitato, seppur molto alto (oltre 150.000 cicli di scrittura).

La memoria di massa: i dischi ottici

Tra i più diffusi troviamo i **CD** (**Compact Disc**), costituiti da un cilindro di plastica policarbonata largo circa 12 cm e alto circa 1,2 mm. Su di esso è incisa una lunga traccia spirale (diversamente dai dischi magnetici organizzati in tracce concentriche suddivise in settori) che parte dal centro e arriva all'esterno del disco ed è formata da una sequenza di aree piane **(land) .** Durante la creazione di un CD, la traccia viene deformata con piccolissimi buchi **(bump)** creati sulle aree piane. Creando i bump sulla spirale non si fa altro che scrivere i singoli bit di ogni byte, 0 (land) e 1 (bump) . Una volta che il pezzo di policarbonato è stato inciso con milioni di bump, uno strato di alluminio riflettente viene stampato per coprirli e proteggerli. Uno strato di acrilico e infine l'etichetta completa il tutto.

Per quanto riguarda la lettura, i bit «ottici» sono letti da una testina che emette un fascio di luce (**laser**) che , durante la rotazione , colpisce le singole aree. Il bump si comporta come uno specchio, e quindi la luce incidente viene riflessa e raccolta da un dispositivo sensibile alla luce **(diodo fotorilevatore) .** Il land, invece , cioè l'area piana, non si comporta come uno specchio e, quindi, la luce è diffusa dai rilievi in tutte le direzioni e, pertanto, non viene rilevato alcun segnale riflesso.

Il DVD (Digital Versatile Disc)-ROM esteriormente è analogo al CD-ROM , ma può contenere da 1,4 a 17 GB (cioè fino a 25 volte la capacità di un normale CD).



Periferiche di input ed output

Un componente hardware è definito anche ***device*** (**periferica**), in quanto è un dispositivo collegato al computer e dipendente da esso. Il device tratta le informazioni di ***input*** (**ingresso**) oppure di ***output*** (**uscita**); alcuni device possono trattare entrambe le informazioni (**input-output** o I**/O**).

La tastiera (keyboard) è il più tradizionale e diffuso dispositivo per l'immissione dei dati (input). La tastiera standard è composta da 102 tasti, che possono essere classificati in quattro modi:

I tasti standard, posizionati nella parte centrale della tastiera, sono disposti quasi nello stesso ordine di quelli di una macchina per scrivere (formato QWERTY) e servono per digitare lettere, vocali accentate , numeri, segni d'interpunzione e gli operatore relazionali e aritmetici.

I tasti numerici servono per la digitazione delle cifre numeriche e sono poste nel settore di destra della tastiera. Per attivare il tastierino numerico occorre premere il pulsante **Boc Num.**

I tasti di controllo si adoperano per svolgere particolari funzioni di servizio generale, particolarmente nelle fasi di battitura dei testi (funzione di spostamento cursore, avanzamento pagine, ecc).

I **tasti funzione** o soft keys, caratterizzati dalle sigle F1 , F2,…,F12 e posti nella parte alta della tastiera, permettendo di effettuare determinate operazioni a seconda del tipo di software utilizzato. Una tastiera può essere collegata al computer tramite il cavo Usb, PS2 o addirittura ad infrarossi o wireless.

Periferiche di input ed output

Il **mouse** fa parte di una particolare tipologia di periferiche, dette **dispositivi di puntamento** (input). Ne esistono di diversi tipi, i più comuni sono i mouse a infrarossi, senza cavo di collegamento al computer. Analoga al mouse è la **trackball**; in questo dispositivo la scatoletta è fissa e il puntatore sul video viene spostato ruotando direttamente una sfera. Un altro dispositivo di puntamento è il **touchpad** presente sul computer portatile. Il touch pad è un dispositivo sensibile al tatto: il movimento del dito sulla sua superficie fa muovere il puntatore sullo schermo.

Lo scanner **(input)** è un dispositivo simile alla comune fotocopiatrice: permette di leggere fogli che contengono testi o disegni, nonché immagini su qualsiasi tipo di supporto fotografico trasformando il tutto in formato digitale. Esistono programmi di tipo **OCR (Optical Character Recognition)** che possono trasformare l'immagine digitale di un testo in un vero e proprio file di testo.

Un altro dispositivo di input per l'acquisizione delle immagini è la **macchina fotografica digitale**, che salvano l'immagine o il filmato su una scheda di memoria esterna (**memory card**). Analogo discorso vale per le **videocamere digitali** (input) che consentono l'acquisizione di immagini e filmati digitali: appartengono a questa categoria anche le **Web Cam** usate per il trasferimento dei video attraverso la rete.

La tavoletta grafica (digitizer) , altro dispositivo di input**,** è composta da un piano di lavoro e da una penna. Il disegno fatto sul piano viene riprodotto, automaticamente , sullo schermo. Può essere utilizzata anche come un mouse.

Il **microfono** è una periferica di input che viene generalmente collegata alla scheda audio del computer e permette di registrare il suono in formato digitale.

Periferiche di input ed output

Il monitor è un dispositivo di output ed è costituito da cristalli liquidi (LCD:Liquid Crystal Display), analoghi a quelli dei computer portatili , differiscono dai monitor CRT in quanto non hanno il tubo catodico e sono di dimensioni più ridotte rispetto alla loro profondità. Ogni punto dello schermo è chiamato pixel (picture element) e costituiscono una caratteristica fondamentale dei monitor perché definiscono la loro risoluzione: maggiore è il numero di pixel visualizzabili, maggiore è la definizione del video. Per esempio, se la risoluzione massima del video è 1024 x 768 , sul monitor possono essere rappresentati al massimo 1024 pixel in orizzontale per 768 pixel in verticale, per una matrice completa di circa 786.432 pixel visualizzabili sullo schermo. La grandezza del pixel è detta dot pitch.

Quanto più sofisticata è l'immagine rappresentata, maggiore sarà la quantità di memoria necessaria per elaborare tale immagine. Nella memoria è riservato un buffer , la mappa video, nel quale vengono memorizzati i codici dei caratteri da visualizzare e la corrispondenza posizione del video.

La stampante è un dispositivo di output e stampano con risoluzioni misurate in DPI (Dots Per Inch, cioè punti per pollice). Le stampanti si classificano in :

Stampanti a impatto, che imprimono il carattere sulla carta i modo meccanico attraverso la pressione del mezzo di scrittura sul nastro inchiostrato .

Stampanti a non impatto (getto di inchiostro , laser e così via) sono più silenziose e raggiungono prestazioni elevatissime. La velocità di stampa è misurate in CPS (caratteri per secondo) oppure in pagine per minuto (ppm).

Il Plotter è un dispositivo utilizzato per applicazioni di carattere tecnico – progettuale. Il formato del foglio può essere A2 (4 volte un A3), A1, A0 o

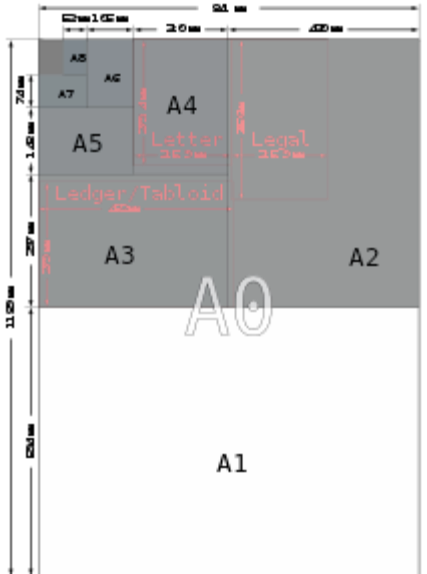
Periferiche di input ed output

Il plotter

È un dispositivo di output utilizzato per applicazioni di carattere tecnico-progettuale. È il dispositivo di output ideale per i sistemi CAD, dove è impiegato per la stampa di prospetti e progetti architettonici, meccanici, elettrici, mappe topografiche, curve geometriche ecc. Oggi viene anche utilizzato nell'ambito della grafica e della pubblicità grazie alle moderne tecnologie che consentono al plotter di stampare a colori e addirittura di ritagliare (plotter da taglio). Il nome deriva dal verbo inglese *to plot* nel senso di tracciare (un diagramma). È composto da una serie di pennini, collegati a un braccio meccanico che si muove orizzontalmente nei due sensi, permettendo, così, di arrotolare e srotolare la carta in base alle esigenze del grafico da realizzare.

La dimensione del supporto di stampa può andare dall'A4 o meno fino a rotoli larghi 914 mm o più. Ad oggi la tipologia più diffusa è di gran lunga quella del plotter a getto di inchiostro con larghezza di stampa di 914 mm, in grado di stampare formati fino al formato "A0+", tali dispositivi possono stampare su fogli singoli oppure su rotoli di carta di cui procede automaticamente al taglio ultimata la stampa.

Formato carta



Periferiche di input ed output

Le casse acustiche

Sono composte da un insieme di altoparlanti (speaker) che permettono l'ascolto dei suoni. La scelta delle casse deve avvenire in funzione della scheda audio presente sul PC: se la scheda audio comprende solo le funzioni di base si useranno casse economiche, se invece è capace di riprodurre audio di alta qualità, effetti audio particolari, come il Dolby surround, si potranno usare casse di qualità superiore.

Gli auricolari e le cuffie

Un auricolare è un piccolo altoparlante che si appoggia all'orecchio, o si inserisce in parte nell'orecchio (in questo caso si parla di auricolare in-ear). Normalmente gli auricolari sono a coppie, così da premettere anche l'ascolto stereofonico.

Una **cuffia** è costituita da una coppia di altoparlanti, generalmente (ma non sempre) più grande degli auricolari, uniti da un supporto, e ha la stessa funzione di una coppia di auricolari. Tutti i computer portatile, i PDA e gli smartphone possiedono apposite uscite (porte) per collegare auricolari e cuffie.

Il monitor touch screen o schermo tattile è un monitor che può rilevare la presenza e la locazione di un tocco o contatto all'interno dell'area visualizzata. Tale tocco può avvenire attraverso il dito di una mano oppure attraverso un altro oggetto passivo, per esempio una penna.

Il Modem è un dispositivo sia di input sia di output che rende possibile la comunicazione fra computer attraverso linee analogiche come quelle telefoniche tradizionali. Proprio perché le linee sono analogiche, progettate per trasformare segnali vocali, è necessario utilizzare un modem, che ha il compito di trasformare i dati provenienti dal computer in un segnale di tipo audio, che viene trasmesso lungo la rete telefonica. Allo stesso tempo, in ricezione, deve essere in grado di eseguire la trasformazione inversa. Il modem prende il nome da questa doppia attività, perché deriva dalla parole inglesi **modulator** e **demodulator**. La velocità del modem può essere misurata in **baud,** cioè il numero di bit trasmessi al secondo oppure in bps (Bit per secondo).I modem attuali Modem ADSL (da 640 kbit/s a 100 Mbit/s) e Modem GPRS/EDGE/UMTS/HSDPA (integrati nei telefonini di ultima generazione o anche come PC card o modem USB, con velocità da 56 kbit/s a 28.8 Mbit/s)

Software di sistema e applicativi

Esistono due tipi di programmi per il computer, sostanzialmente diversi tra loro: il software di sistema e il software applicativo. Il software di sistema, detto anche sistema operativo, si occupa di controllare l'uso di risorse hardware del computer, come la memoria centrale o il disco fisso, e gestisce l'interfaccia di comunicazione tra queste risorse e l'utente. Il SO è il software che gestisce le risorse (hardware e software) della macchina e rende possibile la gestione, l'elaborazione e l'immagazzinamento dell'informazione.

Esistono diversi tipo di sistema operativo, ognuno con proprie caratteristiche e peculiarità. I principale sistemi operativi per personal computer sono:

Windows della Microsoft, **MacOS** per i computer della Apple, Gnu/**Linux** di tipo Open Source. Per i dispositivi mobili i più diffusi sono **Android** e **Windows** per prodotti Samsung, Asus, ecc, **iOS** per i prodotti della Apple.

Funzioni del Sistema Operativo

- Gestione dei Processi *(multitaskin*g)
- Gestione della memoria *(schedule*r)
- Gestione dei files *(file syste*m)
- Gestione degli utenti *(multiutenz*a)
- Gestione dell' I/O *(stampanti, video, tastiera, ...*)
- Gestione servizi di Rete
- Protezione del sistema
- Interprete di comandi
- Altre funzioni di sistema

Tipologie di Sistemi Operativi

- Singolo utente / Multi utente
- Real time
- Sistemi distribuiti
- Embedded (sistema integrato)

Definizione di Processo

Un Processo è un programma in esecuzione. Un processo incorpora le istruzioni, i dati da elaborare, lo stato dell'elaborazione *(contex*t)

Abbiamo due famiglie di sistemi:

-MULTITASKING: processo tramite il quale il sistema operativo può realizzare più operazioni che sembrano contemporanee ma che in realtà avvengono in tempi diversi. Questo lasso di tempo, grazie alla velocità del processore è minimo, per tanto si ha la sensazione che le operazioni si effettuino nello stesso istante.

-SCHEDULER: dispositivo che si prefigge di sostituire un processo, finito o meno, sostituendolo con un altro poiché il sistema operativo permette di lavorare con un solo processo alla volta.

-INTERRUPTS: è un segnale di avvertimento che viene mandato al processore per indicare che qualcosa è avvenuto .

Il **software applicativo** è costituito da programmi che danno al computer la capacità di svolgere compiti particolari e ben precisi. La suite dei software di Office, come **Excel, Powerpoint, Word, Access, Outlook, Visio, Project** ed altri ancora, sono software applicativi ed ognuno di essi svolge un lavoro preciso. Sono software applicativi anche i Browser che visualizzano le pagine web di internet (**Firefox**, **Chrome, Internet Explorer, ecc.)** Appartengono alla categoria dei software applicativi i programmi per fare videoconferenze, antivirus, social network come **Facebook, MySpace o Twitter.** Esistono poi innumerevoli applicazioni per usi specifici come i programmi di elaborazione del suono, i programmi applicativi per il montaggio dei film, delle immagini e software per il disegno. Le **app** sono applicazioni realizzate specificatamente per i dispositivi mobili, per esempio album fotografici, calendari, strumenti di comunicazioni e giochi.

Software proprietario, freeware, shareware e open source.

Alcuni sviluppatori di applicazioni software distribuiscono liberamente i propri programmi, anche attraverso internet, senza chiedere alcun compenso: si parla in tal caso di **freeware.** Spesso il software commerciale si può usare gratuitamente in prova per un periodo, dopo di che non funziona più a meno che lo si acquisti. Nel caso del **shareware**, l'accordo è che dopo il periodo di prova, se non si vuole acquistare il software, lo si deve rimuovere dal proprio computer; continuare a usare un'applicazione shareware anche dopo il periodo di prova è illegale. Le istruzioni di un programma o codice-sorgente (source code) sono inaccessibili a chi usa un software proprietario: in questo modo gli sviluppatori proteggono l'originalità della propria creazione, impedendo agli altri di apportare modifiche. Differentemente il software Open Source è aperto e quindi modificabile, tale principio di libera collaborazione è oggi in continua condivisione tra i migliori specialisti del mondo.

Informatica e società

La scalata degli strumenti informatici.

La grandissima diffusione degli strumenti informatici ha modificato , migliorato e velocizzato la vita quotidiana di ciascuno di noi. Basti pensare che la tecnologia informatica viene, ormai, efficacemente e normalmente impiegata in molteplici apparecchiature, dagli ascensori intelligenti, ai normali elettrodomestici. La versatilità del computer insieme con la sua elevata potenza elaborativa consente di soddisfare le richieste e le esigenze di moltissima gente. Sono questi i motivi che ne hanno decretato il successo nel corso degli ultimi decenni e hanno dato vita all'**ICT** (**Information and Communication Technology**).

Esistono dei compiti , ormai testati, per i quali il computer può sostituire l'uomo totalmente, il cui intervento è richiesto solo in caso di emergenza.

Ne sono esempi caratteristici le seguenti attività:

- La gestione di centrali telefoniche;
- L'automazione delle linee di produzione;
- Il controllo e la gestione di strumenti di laboratorio;
- Il monitoraggio ambientale;
- Il monitoraggio di situazioni critiche (ospedali, sale di terapia intensiva, ecc).

Il computer ha portato nel corso del tempo un notevole contributo al miglioramento della qualità della vita delle persone diversamente abili, con strumenti quali:

- Tastiere Braille per computer;
- Stampanti di documenti in codice Braille;
- Lettori di schermi video mediante polpastrelli;
- Controllo del computer tramite la voce;
- Comando di automatismi per attivazioni di allarmi, apertura porte, accensione e spegnimento luci, attraverso sistemi computerizzati;
- Riconoscimento vocale per la dettatura di documenti ed altro ancora.

Divertimento e studio

I maggiori fruitori delle molteplici capacità di un computer sono i ragazzi di ogni età che lo usano soprattutto per giocare. Esistono, però, programmi di intrattenimento che esercitano le capacità logiche dei ragazzi. Il computer , infatti, può essere utilizzato per la didattica . L'istruzione assistita dal computer (**CBT , Computer Based Training**) è ormai una realtà consolidata e i software che rispondono a questa finalità sono utilizzati sia in ambito scolastico sia in azienda per la formazione del personale, anche a distanza. Questa metodologia è conosciuta con il termine **e-learning** che indica, appunto, l'insegnamento e l'apprendimento in rete. Lo studente può seguire l'insegnante collegato in rete ed in tempo reale, grazie a internet, e seguire le lezioni a distanza (**modo sincrono**) , oppure scaricare la lezione ed eseguire i test in momenti diversi (**modo asincrono**).

Con il passare del tempo e con l'avvento di computer sempre più potenti e delle reti telematiche, si sta optando per una nuova forma di commercio : quella identificata dal neologismo **e-commerce** (commercio elettronico).

Le più diffuse forme di commercio elettronico sono denominate dalla sigla Businnes to Businnes (B2B), Businnes to Consumer (B2C), Consumer to Consumer (C2C) e Consumer to Businnes (C2B) dove, in questo ultimo caso, sono i consumatori stessi ad offrire ad un determinato prezzo i prodotti/servizi alle aziende.

Il computer in banca

I servizi bancari vengono offerti attraverso:

- Sportello tradizionali, dotati di computer;
- Servizi di home banking (accesso da casa tramite computer e internet);
- Terminali self-service (servizio bancomat)

Il computer nella Pubblica Amministrazione

Uno dei settori che, negli ultimi anni, ha sviluppato in maniera efficace i servizi telematici ai cittadini è quello della Pubblica Amministrazione. Molti di questi servizi vengono erogati attraverso sportelli self-service **(e-government)** che consentono di:

- Velocizzare il rilascio di certificati e documenti;
- Semplificare l'accettazione di domande;
- Semplificare il pagamento di tasse e contributi;
- Fornire informazioni sui servizi

L'ergonometria

L'utilizzo del computer, soprattutto se prolungato , può provocare qualche disturbo, principalmente per l'apparato muscolo-scheletrico e per la vista, o problemi di affaticamento mentale. Tuttavia, osservando alcune norme di buona pratica è possibile prevenire.

Sottolineiamo che le informazioni sui rischi a cui sono esposti colore che utilizzano abitualmente il computer, sono raccolte e trattate dal **decreto legislativo 626/94: Allegato VII** (Prescrizioni minime) e dal **decreto ministeriale 2 ottobre 2000** (linee guida d'uso dei videoterminali). Nello specifico del lavoro al computer le attuali norme in vigore prescrivono l'obbligo di valutare l'ergonomia dei siti di lavoro e renderli adeguati alle idonee condizioni ergonomiche.

Il Piano di lavoro ottimale della scrivania deve avere una profondità sufficiente per disporre il monitor alla corretta distanza visiva (tra i 50 e 70 cm) e deve consentire l'appoggio per gli avambracci davanti alla tastiera, durante la digitazione. Nei limiti del possibile sono da evitare i tavolini da computer con la tastiera disposta su un piano estraibile se questo non permette l'appoggio degli avambracci.

L'ergonometria

La scrivania deve essere stabile e di altezza, fissa o regolabile, indicativamente fra 70 e 80 cm; con uno spazio sufficiente per garantire una posizione comoda e libertà di movimento delle gambe; avere una superficie preferibilmente di colore chiaro e in ogni caso non riflettente.

Il sedile deve essere stabile, antiribaltamento, deve permettere una certa libertà di movimento, meglio se girevole, con altezza regolabile, schienale regolabile in altezza e inclinazione e buon appoggio lombare per una posizione comoda.

La posizione corretta

Per prevenire disturbi muscolo- scheletrici è importante assumere una posizione corretta di fronte al monitor, con piedi ben poggiati al pavimento, angolo di 90 gradi tra cosce e busto, schiena dritta ben appoggiata allo schienale della sedia nel tratto lombare, regolando l'altezza della sedia e l'inclinazione dello schienale a circa 90 gradi rispetto al movimento.

Posizionare lo schermo del monitor di fronte evitando se possibile posizione a 45 gradi che obbligano il collo a una posizione innaturale. Regolare il monitor in modo che lo spigolo superiore dello schermo sia posto leggermente più in basso della linea degli occhi.

Evitare , per quanto possibile, posture fisse per tempi prolungati, cambia di tanto in tanto la posizione, muoviti per rilassare i muscoli e pratica esercizi di stretching per il collo, schiena , braccia e gambe.

I virus

Il termine virus è entrato amaramente a far parte del vocabolario di chi utilizza un computer. Ma cos'è esattamente un virus?

Un virus informatico è un piccolo programma che contiene alcune istruzioni addette alla replicazione dell'intero programma. Una volta replicato, il virus inizia a svolgere attività che possono essere distruttive e/o di ostruzionismo.

Tipi di virus

E' in realtà più opportuno e rigoroso parlare di codice malefico riferendosi a questi programmi dalla potenza distruttiva che si replicano autonomamente. Vediamo alcuni tipi di virus:

- **Bootsector**: si annidano di solito nei supporti rimovibili (per esempio pendrive). In genere sono molto pericolosi: possono impedire al sistema operativo di prendere il controllo del computer costringendoci alla formattazione o possono cancellare file importanti.
- **Worm** (vermi) : sono alquanto pericolosi e abbastanza diffusi. Cercano parti inutilizzate dell'hard disk per riprodursi, fino a portare al collasso del sistema. Si propagano anche per mezzo della posta elettronica (e-mail worm).
- **Trojan:** prendono il nome dal mitico cavallo di Troia. Sono usati per creare una **falla** nel sistema, attraverso la quale il loro programmatore può penetrare nel computer infetto anche se questo è protetto da sistemi di sicurezza. Non si possono considerare veri e propri virus , ma risultano particolarmente fastidiosi. Con questo criterio funzionano anche le famose **bombe logiche,** che si attivano in determinati giorni dell'anno.
- **Spyware:** si tratta di un programma che s'installa nel computer e raccoglie informazioni sulla attività online. Diversamente da virus e worm , non è in grado di diffondersi autonomamente. E' invece più simile ad un trojan , poiché necessita dell'intervento dell'utente per essere installato. I danni provocati possono andare dalla semplice violazione della privacy, che si traduce nell'invio di pubblicità mirata tramite finestre nel browser o attraverso e-mail di spam, al più grave furto dei tuoi dati sensibili.

Sniffing

Si definisce sniffing l'attività di intercettazione passiva dei dati che transitano in una rete telematica. Tale attività può essere svolta sia per scopi legittimi (ad esempio l'individuazione di problemi di comunicazione o di tentativi di intrusione) sia per scopi illeciti (intercettazione fraudolenta di password o altre informazioni sensibili).

I prodotti software utilizzati per eseguire queste attività vengono detti sniffer ed, oltre ad intercettare e memorizzare il traffico, offrono funzionalità di analisi del traffico stesso. Gli sniffer intercettano i singoli pacchetti, decodificando le varie intestazioni di livello datalink, rete, trasporto, applicativo. Inoltre possono offrire strumenti di analisi che verificano ad esempio tutti i pacchetti di una connessione TCP per valutare il comportamento del protocollo o per ricostruire lo scambio di dati tra le applicazioni.

Sniffing del traffico locale

Il traffico può essere intercettato da uno degli host coinvolti nella comunicazione, indipendentemente dal tipo di interfaccia di rete su cui viene inviato.

Sniffing in reti locali

Per intercettare i dati in una rete locale è necessario possedere od ottenere l'accesso fisico al mezzo trasmissivo.

La tutela dei dati personali tra DPS e modelli organizzativi

Tutti sappiamo che esiste, ormai da diverso tempo, quella che viene comunemente chiamata "**legge sulla privacy**", ma pochi di noi sono davvero informati sugli effetti pratici di questa normativa. Le finalità del d. lgs. n. 196/2003 consistono nel riconoscimento del diritto del singolo sui propri dati personali e, conseguentemente, nella disciplina delle diverse operazioni di gestione (tecnicamente "trattamento") dei dati, riguardanti la raccolta, l'elaborazione, il raffronto, la cancellazione, la modificazione, la comunicazione o la diffusione degli stessi.

Il diritto assoluto di ciascuno sui propri dati è esplicitamente riconosciuto dall'art. 1 del testo unico, in cui si afferma: **"Chiunque ha diritto alla protezione dei dati personali che lo riguardano".** Tale diritto appartiene alla categoria dei diritti della personalità. Il diritto sui propri dati è differente dal diritto alla riservatezza, in quanto non riguarda solamente informazioni inerenti la propria vita privata, ma si estende in generale a qualunque informazione relativa ad una persona, anche se non coperta da riserbo (sono dati personali ad esempio il nome o l'indirizzo della propria abitazione).

Lo scopo della legge non è quello di impedire il trattamento dei dati, ma di evitare che questo avvenga contro la volontà dell'avente diritto, ovvero secondo modalità pregiudizievoli. Infatti definisce i diritti degli interessati, la modalità di raccolta e i requisiti dei dati, gli obblighi di chi raccoglie, detiene o tratta dati personali e le responsabilità e sanzioni in caso di danni.

La tutela dei dati personali tra DPS e modelli organizzativi

Cosa sono i dati personali ?

Per poter comprendere le regole a protezione della privacy occorre chiarire cosa si intende per **"dato personale".** Tale aspetto è infatti essenziale per comprendere le regole ed applicarle correttamente. Secondo la normativa, il dato personale è qualunque informazione relativa a persona fisica, persona giuridica, ente od associazione, identificati o identificabili, anche indirettamente, mediante riferimento a qualsiasi altra informazione, ivi compreso un numero di identificazione personale. Dato personale è però anche un'immagine, un suono e qualunque notizia o informazione che sia riferibile a un soggetto determinato o determinabile.

I codici identificativi, sia quelli ricavati da dati anagrafici (ad esempio il **codice fiscale**), che i codici univoci attribuiti a una persona in base a criteri predefiniti (ad esempio i **codici cliente**) sono dati personali. Dato personale è quindi qualsiasi informazione riferita (o anche semplicemente riferibile tramite un codice) a una persona, anche il numero di targa di una vettura riferita a un proprietario o il numero di una polizza riferita a un assicurato.

Una categoria particolare di dati personali sono i **dati sensibili.** Si tratta dei dati personali idonei a rivelare l'origine razziale ed etnica, le convinzioni religiose, filosofiche o di altro genere, le opinioni politiche, l'adesione a partiti, sindacati, associazioni od organizzazioni a carattere religioso, filosofico, politico o sindacale, nonché i dati personali idonei a rivelare lo stato di salute e la vita sessuale.

Questa tipologia di dati è sottoposta a un livello di protezione più elevato di quello previsto per i dati non sensibili.
La prima verifica che **il titolare del trattamento dei dati** deve eseguire, consiste nell'accettarsi della tipologia di dati trattati dal sistema informativo aziendale.
Qualora, da questa indagine dovesse emergere informazioni tali da poter affermare che l'organizzazione tratta dati sensibili o addirittura giudiziari , si dovranno seguire delle politiche particolari denominate **policy sui dati sensibili**. Le policy sui dati sensibili sono documenti che orientano l'utente nell'elaborazione, nell'accumulazione e nella trasmissione di dati sensibili. Il Codice sulla protezione dei dati personali (d.lgs. 196/2003), art.4, disciplina e specifica quali dati siano da considerarsi sensibili. Esistono inoltre particolari politiche atte a favorire la sicurezza e la segretezza dei dati nelle procedure di trasmissione on line (Internet Security Policy). Il codice, nella fattispecie, definisce come trattamento di dati personali «qualunque operazione o complesso di operazioni effettuati anche senza l'ausilio di strumenti elettronici, concernenti la raccolta, la registrazione, l'organizzazione, la conservazione, la consultazione, l'elaborazione, la modificazione, la selezione, l'estrazione, il raffronto, l'utilizzo, l'interconnessione, il blocco, la comunicazione, la diffusione, la cancellazione e la distruzione di dati, anche se non registrati in una banca di dati» (art. 4, lett. a, del Codice della Privacy - Dlgs 196/2003).

Privacy e sicurezza informatica

Di crescente rilievo è il tema della sicurezza informatica che riguarda sia i privati cittadini, sia le imprese: esso coinvolge tutti gli aspetti che riguardano la protezione dei dati sensibili archiviati digitalmente ma in particolare è noto al grande pubblico con riferimento all'utilizzo di Internet.

In effetti, la rete è in grado di offrire una vasta gamma di informazioni e servizi ma contemporaneamente può costituire un luogo pericoloso per la nostra privacy anche perché il mezzo stesso non è stato concepito per scambiare o gestire dati sensibili.

In un contesto simile, mantenere l'anonimato risulta spesso arduo e con il proliferare dei conti on-line e lo spostamento delle aziende su Internet, risulta più semplice per i malintenzionati accedere alle nostre informazioni riservate. A tal proposito, una delle piaghe più dannose della rete è lo spyware che, installandosi spesso in maniera fraudolenta nel personal computer delle vittime, provvede ad inviare dati personali (pagine visitate, account di posta, gusti ecc) ad aziende che successivamente li rielaboreranno e rivenderanno.

Esiste perfino un metodo, chiamato social engineering, tramite cui i truffatori riescono a ottenere informazioni personali sulle vittime attraverso le più disparate tecniche psicologiche: si tratta di una sorta di manipolazione che porta gli utenti a rilasciare spontaneamente i propri dati confidenziali.

La miglior difesa per la nostra privacy, in questa situazione di precarietà, consiste nell'utilizzare il buon senso e nell'adottare semplici accorgimenti tra cui:

Evitare il più possibile di comunicare la propria password.

Installare e configurare bene firewall e antivirus tenendoli in seguito costantemente aggiornati.

Procurarsi un antispyware in grado di ripulire efficacemente il sistema.

Non aprire allegati di e-mail provenienti da utenti sconosciuti o sospetti per evitare fenomeni di cosiddetto phishing.

Esistono inoltre soluzioni meno immediate ma più efficaci come l'utilizzo della crittografia, che ci permette di criptare un messaggio privato attraverso particolari software facendo sì che solo l'utente destinatario possa leggerlo in chiaro, unito all'implementazione della firma digitale.

La tutela della privacy **non** può essere attuata tramite l'assunzione di false identità (nella legislazione italiana questo è un reato)

Privacy - Regolamento UE 2016/679

GDPR (General Data Protection Regulation) è il nuovo regolamento sulla protezione dei dati personali (Reg UE 2016/679) che entrerà in vigore il prossimo 25 maggio 2018 e contemporaneamente verrà abrogata la Direttiva 95/46 EU che ha dato origine al nostro Dlgs 196/2003 e alle altre corrispondenti normative privacy europee

L'obiettivo del Regolamento è garantire a tutti i cittadini dell'UE il rispetto dei "diritti e delle libertà fondamentali delle persone fisiche". **Non si parla più di privacy, ma di diritto alla protezione dei dati personali.**

Il Regolamento ha uguali effetti in tutti gli stati dell'UE, unifica i trattamenti dei dati personali e impone nuovi obblighi di conformità. Per garantire la compliance (conformità) al nuovo Regolamento Europeo GDPR. Le aziende e gli enti pubblici dovranno definire, gestire, documentare e attuare alcuni processi che si snodano in modo trasversale nell'intera organizzazione fondato sul concetto anglosassone di *accountability* (responsabilizzazione).

Attività da svolgere per uniformarsi al regolamento:

- **Nomina** del Responsabile della Protezione dei Dati personali
- **Nomina** del Data Protection Officer (DPO); Il DPO è una figura di controllo priva di responsabilità esecutive, in possesso di **competenze ed esperienze specifiche** che esprime solo pareri ai sensi infatti dell'art. 37 Reg. 2016/679 la nomina di un DPO risulta obbligatoria in 3 casi definiti:

  -se il trattamento è svolto da un'autorità pubblica o da un organismo pubblico;
  -se le attività principali del titolare o del responsabile consistono in trattamenti che richiedono il monitoraggio regolare e sistematico di interessati su larga scala;
  -se le attività principali del titolare o del responsabile consistono nel trattamento su larga scala di categorie particolari di dati relativi a condanne penali e reati.
- **Registro** dei Trattamenti e Registro delle Attività dei Trattamenti
- **PIA** (Protection Impact Assessment), valutazione di impatto preventiva relativa ai trattamenti dei dati, finalizzata alla mitigazione dei rischi (c.d. risk analysis);
- **Privacy by Design**, adozione di metodologia di protezione dei dati sin dalla fase di progettazione;
- **Privacy by Default**, adozione di metodologia di tutela delle informazioni più restrittiva possibile;
- **Consenso**, rafforzamento del consenso esplicito al trattamento dei dati
- **Data Breach Notification**, comunicazione all'Autorità Garante nel caso di accessi non autorizzati;
- **Portabilità**, trasferimento delle informazioni da un fornitore ad un altro;
- **Diritto all'oblio**, cancellazione dei dati personali;
- **Accesso**, miglioramento del diritto di accesso, maggiore chiarezza e comprensibilità;
- **Sanzioni**, in caso di gravi violazioni potranno raggiungere il 4% del fatturato mondiale.

La produzione del software

Lo sviluppo di un progetto è un sistema complesso di attività tese ad ottenere un risultato che chiamiamo prodotto. In un progetto compaiono normalmente l'attività di studio, l'attività di ideazione, l'attività di progettazione, l'attività di realizzazione e l'attività di produzione-

Tali attività sono così definite:

Attività di studio	Studia l'area di competenza del progetto per conoscere approfonditamente la materia
Attività di ideazione	Crea l'astrazione del prodotto e ne definisce le caratteristiche.
Attività di progettazione	Consolida l'astrazione del prodotto formalizzandolo tramite disegni, modelli e prototipi.
Attività di realizzazione	Crea realmente i primi esemplari del prodotto.
Attività di produzione	Termina il progetto ed inizia la produzione.

Il prodotto di un progetto informatico è l'insieme dei moduli software (**programmi**) e degli archivi elettronici (**dati**) idoneo a soddisfare gli obiettivi definiti.

Lo sviluppo di progetti informatici, di media e grande complessità, richiede un'organizzazione del lavoro tale da dare la certezza che i programmi e i dati ottenuti siano razionali, completi ed integrati tra loro. Cioè si vogliono ottenere dei risultati di qualità.

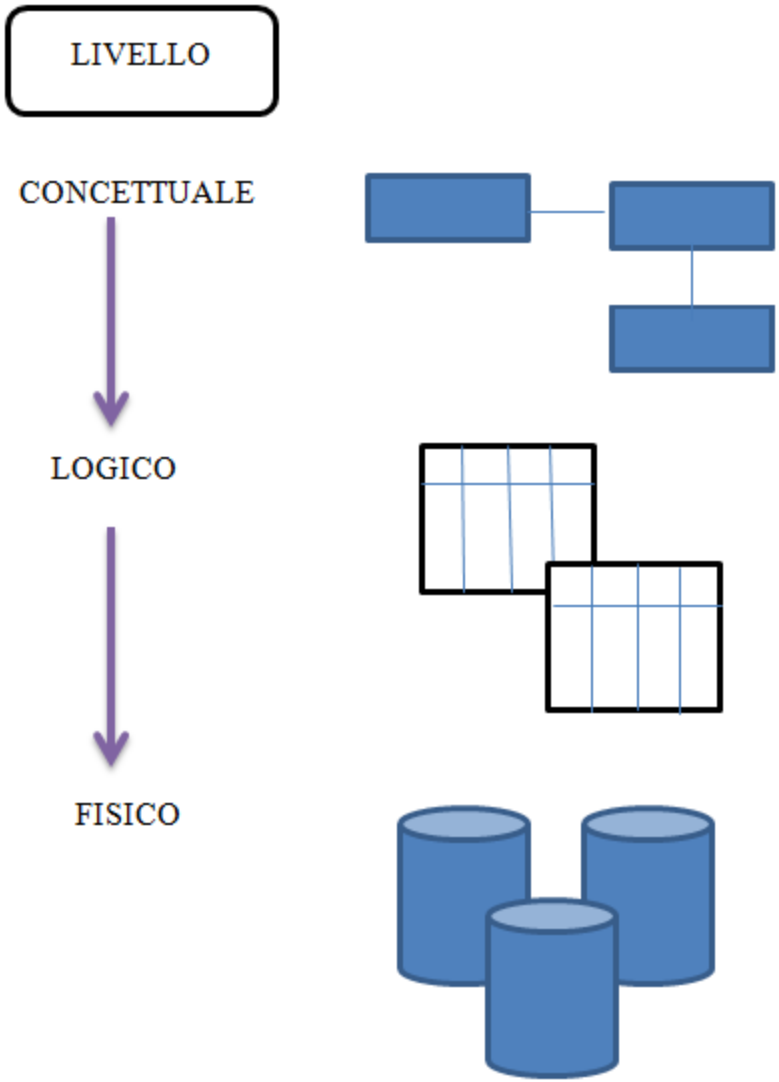
La **qualità di un prodotto** è il grado di aderenza dei risultati rispetto ai bisogni. In altre parole, dobbiamo produrre quello che ci è stato richiesto. Quest'ultima affermazione è all'apparenza banale, ma da non sottovalutare in quanto è particolarmente difficile gestire progetti complessi in modo da ottimizzare il lavoro da svolgere e garantire un prodotto di qualità.

Modellazione dei dati

Modellare i dati significa costruire una rappresentazione semplificata della realtà osservata o di un problema aziendale, individuandone gli elementi caratterizzanti e i legami interconnessi tra essi.

La progettazione di un modello di dati avviene quindi a livelli diversi:

- Il **livello concettuale** (o **esterno**) rappresenta la realtà dei dati e le relazioni tra essi attraverso uno schema.
- Il **livello logico** rappresenta il modo attraverso il quale i dati sono organizzati negli archivi elettronici: descrive quindi la composizione ed il formato dei dati nel loro aspetto di struttura logica di dati. Il livello logico viene derivato dal livello concettuale applicando alcune regole molto semplici.
- Il **livello fisico** rappresenta l'effettiva installazione degli archivi elettronici: esso indica l'ubicazione dei dati nelle memorie di massa (dischi). Il livello fisico è quindi l'implementazione del livello logico sui supporti per la registrazione fisica dei dati: partizioni, puntatori, blocchi fisici, cluster, indici.


LIVELLO
CONCETTUALE
LOGICO
FISICO

L'attività di progettazione consente prima di tutto di costruire una rappresentazione astratta della realtà in modo indipendente dalla struttura dei dati.
Il modello concettuale viene definito attraverso lo **schema** dei dati, cioè una rappresentazione sintetica (di solito presentata in forma grafica) degli elementi fondamentali che caratterizzano la realtà osservata. Questa rappresentazione è indipendente da:

- i valori che verranno assegnati ai dati;
- le applicazioni degli utenti che utilizzano i dati;
- le visioni parziali dei dati da parte degli utenti.

Il modello concettuale rappresenta un patrimonio importante per le aziende, poiché descrive i dati esistenti in azienda: il suo valore informativo può essere utilizzato sia nel campo informatico sia nell'ambito gestionale e diventa un supporto per i diversi ruoli aziendali.
Con il passaggio al modello logico, l'insieme dei dati viene dotato di una struttura che deve facilitare:

- la **manipolazione** o il trattamento dei dati, cioè la possibilità di inserire, modificare e cancellare i dati;
- l'**interrogazione**, cioè la possibilità di ritrovare i dati, richiesti da un'applicazione, in modo semplice e veloce.

Queste strutture di dati vengono poi implementate sulle memorie di massa, realizzando in pratica il modello fisico rappresentato dai file registrati nei blocchi del disco.

IL MODELLO E/R (ENTITA'/ASSOCIAZIONI)

Il modello E/R (Entity/Relationship) è uno strumento per analizzare le caratteristiche di una realtà in modo indipendente dagli eventi che in essa accadono, cioè per costruire un modello concettuale dei dati indipendentemente dalle applicazioni. Viene spesso utilizzato nella prima fase della progettazione di una base di dati in cui è necessario tradurre le informazioni risultanti dall'analisi di un determinato dominio in uno schema concettuale. Il modello E/R si basa su un insieme di concetti molto vicini alla realtà di interesse: quindi facilmente intuibili dai progettisti (e in genere considerati sufficientemente comprensibili e significativi anche per i non-tecnici).

Il risultato è la definizione di una rappresentazione grafica, detta schema E/R, che mette in evidenza gli aspetti fondamentali del modello concettuale con i dati caratterizzanti e le associazioni tra essi.

Gli elementi di un modello entità/associazioni sono:

- entità.
- associazioni,
- attributi.

L'entità

L'**entità** è un oggetto (concreto o astratto) che ha un significato anche quando viene considerato in modo isolato ed è di interesse per la realtà che si vuole modellare.

Esempi di entità sono: una persona, un modello di automobile, un movimento contabile, una prova sostenuta da uno studente.
Le entità possono essere classificate secondo un certo criterio di omogeneità definendo **il tipo di entità** attraverso un nome.
Ciascuna entità ha un nome, per esempio gli studenti di una scuola sono classificabili nel tipo entità Studente. Ciascun studente rappresenta un'**istanza** dell'entità Studente.

Nella rappresentazione grafica le entità sono identificate con un rettangolo contenente all'interno il nome dell'entità.

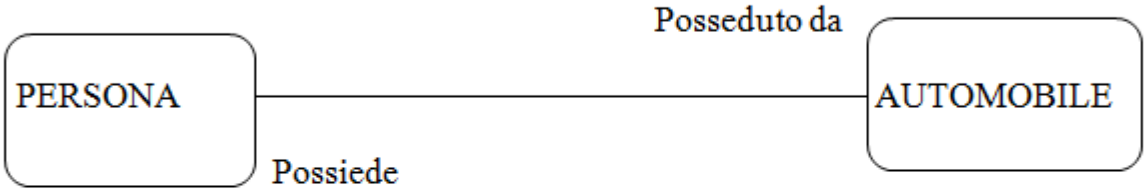
L'associazione

L'**associazione** (in inglese **relationship**) è un legame che stabilisce un'interazione tre le entità. Ogni associazione ha due versi con specifici significati; ogni verso ha un'entità di partenza e una di arrivo; ogni verso inoltre ha una descrizione che consente di comprenderne il significato.

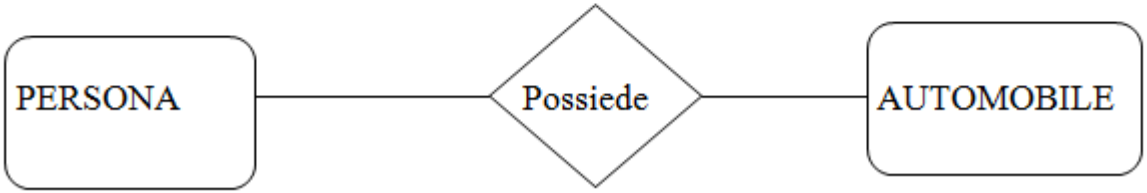
Per esempio tra l'entità Persona e l'entità Automobile esiste un'associazione che può essere descritta nel linguaggio naturale secondo due versi: una persona possiede una o più automobili e un'automobile è posseduta da una persona.

Quindi si può dire che tra l'entità Persona e l'entità Automobile esiste l'associazione *Possiede*; tra l'entità Automobile e l'entità Persona esiste l'associazione *Posseduta da*.

Il simbolo grafico convenzionale usato è la linea, la quale unisce le due entità interessate; la descrizione compare a fianco della linea di partenza del verso.



Un altro simbolismo usato per descriver un'associazione è dato da una linea che unisce le due entità con l'aggiunta, a metà della linea, di un rombo che contiene la descrizione dell'associazione.



L'esempio mostra che solitamente i sostantivi che compaiono nelle frasi del linguaggio naturale corrispondono alle entità, mentre i verbi corrispondono alle associazioni.

Gli attributi

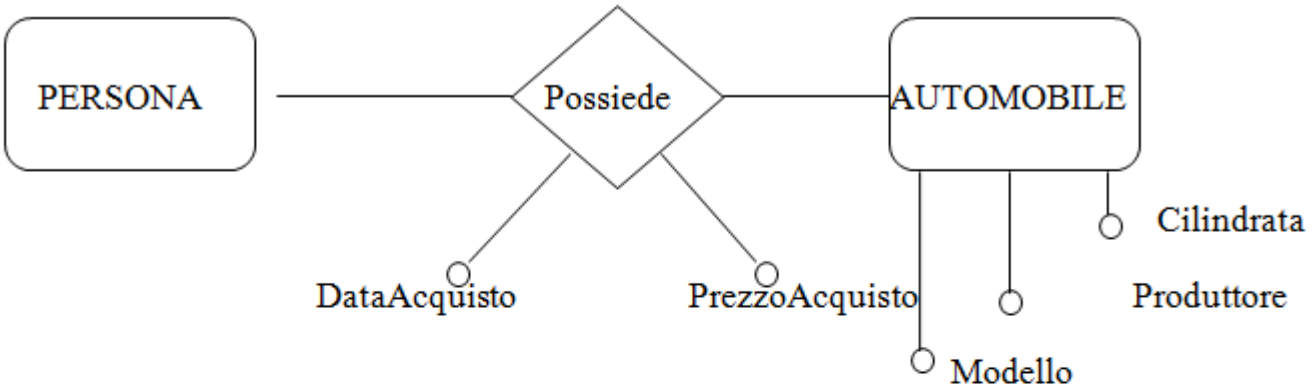
Le proprietà delle entità e delle associazioni vengono descritte attraverso gli attributi. Esempi di attributi per l'entità Automobile sono: Modello, Produttore, Cilindrata, PrezzoListino.

Le **caratteristiche** di ogni attributo sono **il formato, la dimensione e l'opzionalità**:

- il **formato** di un attributo indica il tipo di valori che assume; i tre formati base sono carattere, numerico, data/ora;
- la **dimensione** indica la quantità massima di caratteri o cifre inseribili;
- l'**opzionalità** indica la possibilità di non essere sempre valorizzato: l'attributo è obbligatorio se deve avere valor non nullo, facoltativo se sono accettabili valori nulli (Null).

Si chiama dominio dell'attributo l'insieme dei possibili valori assunti da un attributo.

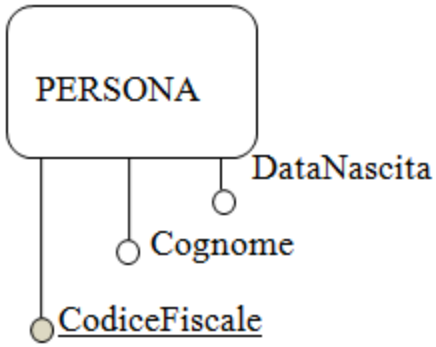
Il simbolo grafico convenzionale usato per rappresentare l'attributo è la linea che parte dall'entità o dall'associazione e termina con il nome ed un piccolo cerchio.



Gli attributi DataAcquisto e PrezzoAcquisto non sono attributi né dell'entità Persona, né dell'entità Automobile, ma sono attributi dell'associazione tra le due entità.

Gli schemi entità/associazioni possono non contenere la rappresentazione grafica degli attributi per evitare un appesantimento nella lettura e nell'interpretazione.

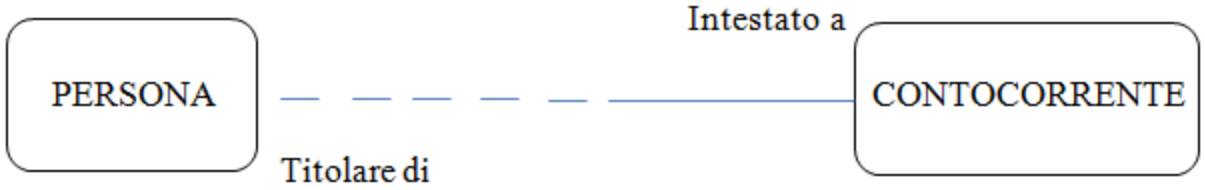
Si indica con il termine **chiave primaria** (**primary key**) l'insieme di uno o più attributi che consentono di distinguere un'istanza dall'altra per la stessa entità: esempi di chiavi sono il codice di un prodotto, la matricola di un dipendente, la chiave composta dal codice studente insieme alla data e al codice della materia per le prove scolastiche. Nella descrizione grafica, gli attributi chiave vengono evidenziati sottolineandone il nome oppure colorando il cerchietto dell'attributo.



Le associazioni tra entità

Un'associazione tra le entità può essere obbligatoria oppure opzionale: **obbligatoria** quando il legame tra le entità deve essere sempre presente, **opzionale** quando può essere presente. Queste caratteristiche possono naturalmente riguardare anche i versi dell'associazione tra le entità.

Per esempio nell'associazione tra l'entità PERSONA e l'entità CONTOCORRENTE, il verso Titolare di è opzionale, perché non tutte le persone possiedono un conto corrente, e il verso Intestato a è obbligatorio, perché ogni conto corrente deve essere intestato a qualcuno. Il simbolismo che diversifica il verso obbligatorio è la linea continua , mentre la linea tratteggiata indica l'opzionalità.



Il **grado** di un verso dell'associazione è la caratteristica che indica quante istanze dell'entità di arrivo si associano all'istanza dell'entità di partenza.

Il grado può essere **a uno** oppure **a molti** e pertanto le associazioni tra due entità si classificano nei seguenti tipi:

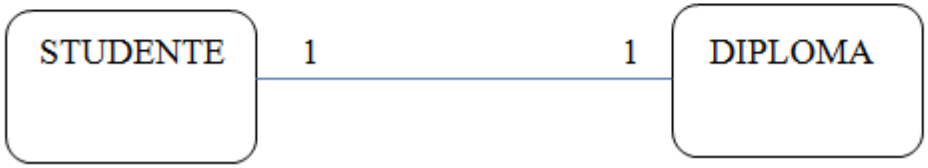
a. Associazione 1:1 (uno a uno) o biunivoca

Ad ogni elemento del primo insieme E_1 corrispondono uno e un solo elemento del secondo insieme E_2 , e viceversa.

Ogni istanza della prima entità si deve associare ad una sola istanza della seconda entità e viceversa.

Per esempio l'associazione tra l'entità Studente e l'entità Diploma, in una scuola superiore, è univoca perché ad ogni studente corrisponde uno e un solo diploma.

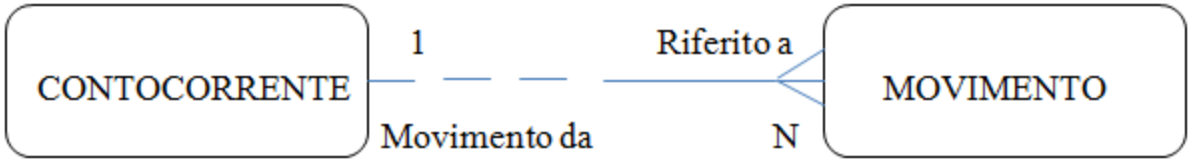
Il simbolismo che indica il grado a uno nell'associazione tra le entità è la line stessa.



b. Associazione 1:N (uno a molti) o semplice

Ad un elemento di E_1 possono corrispondere più elementi di E_2, mentre ad ogni elemento di E_2 deve corrispondere uno e un solo elemento di E_1.

Ogni istanza della prima entità si può associare a una o più istanze della seconda entità, mentre ogni istanza della seconda entità si deve associare ad una sola istanza della prima.
Per esempio nella gestione dei movimenti di un conto corrente, ogni conto può effettuare una o più operazioni, ma ogni movimento deve riferirsi a un solo conto corrente. Quindi nell'associazione tra l'entità ContoCorrente e l'entità Movimento, il verso Movimentato da (opzionale) è di grado a molti, il verso Riferito a (obbligatorio) è di grado a uno.
Il grado a molti si rappresenta con l'aggiunta di altre due linee in prossimità dell'entità di arrivo.

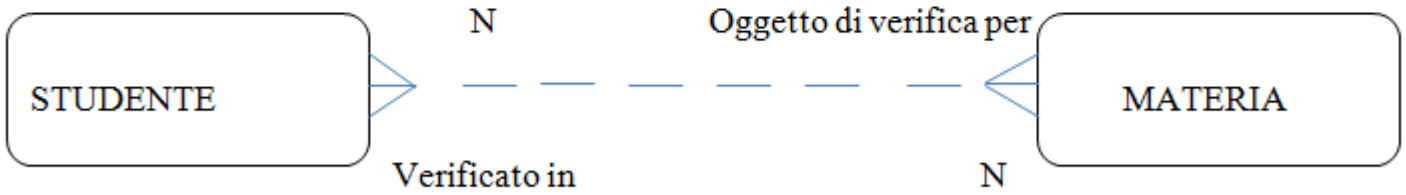


c. Associazione N:N (molti a molti) o complessa

Ad ogni elemento dell'insieme E_1 possono corrispondere più elementi dell'insieme E_2 e viceversa.

Ogni istanza della prima entità si può associare a una o più istanze della seconda entità e viceversa.

Per esempio nell'associazione tra l'entità Studente e l'entità Materia, uno studente può essere verificato su una o più materie, e una materia può essere oggetto di verifica di uno o più studenti.

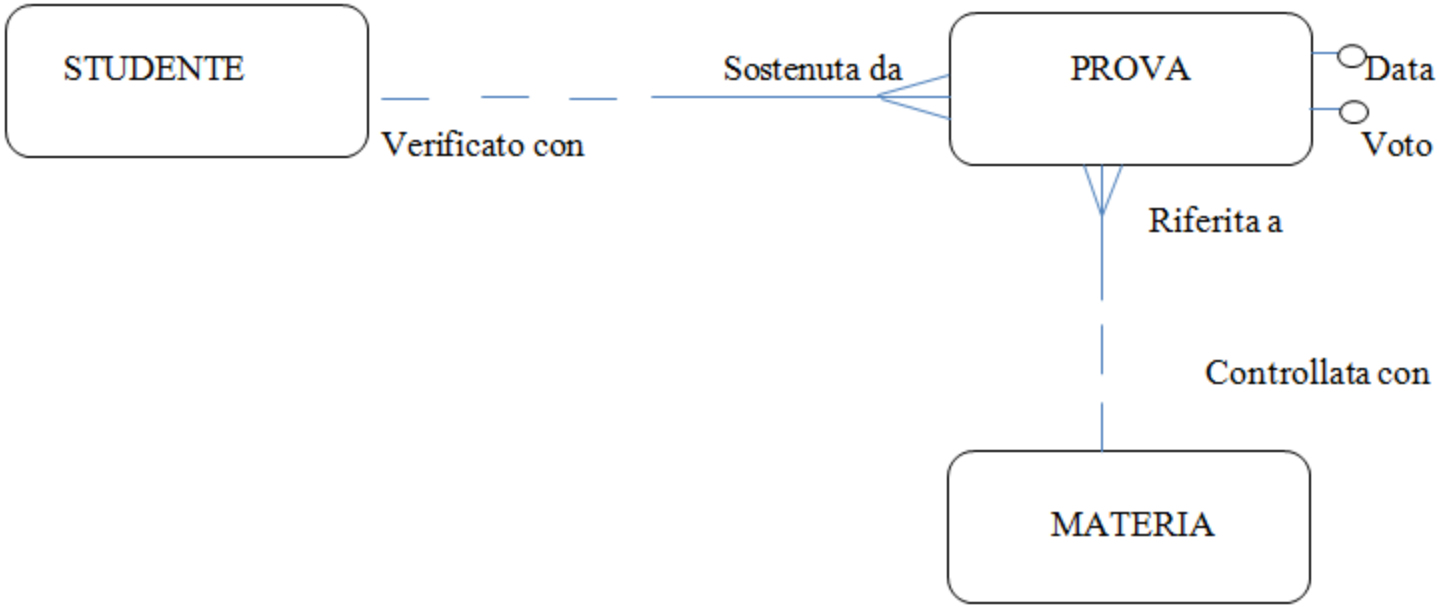


Occorre osservare che nella modellazione dei dati con l'approccio Entity/Relationship le associazioni 1:N sono molto più frequenti delle associazioni di tipo uno a uno e molti a moti.

L'associazione molti a molti può essere facilmente scomposta in due associazioni uno a molti , anche per consentire di rappresentare gli attributi dell'associazione-

Per esempio nell'associazione tra l'entità Studente e l'entità Materia la data della prova e il voto ottenuto dallo studente non sono attributi né dell'entità Studente, né dell'entità Materia.

Il modello diventa più chiaro introducendo una terza entità Prova, avente come attributi la data e il voto della verifica.



L'associazione tra Studente e Prova è di tipo 1:N, in quanto ogni studente può essere verificato con una o più prove e ogni prova deve essere sostenuta da un solo studente; l'associazione tra Materia e Prova è di tipo 1:N, perché ogni materia può essere controllata con una o più prove e ogni prova deve essere riferita ad una sola materia.
L'associazione molti a molti è sta trasformata in due associazioni uno a molti , con l'aggiunta di una terza entità.

Le regole di derivazione del modello logico.

Dal modello concettuale dei dati è possibile ottenere il **modello logico** dei dati; in altre parole si può definire la struttura degli archivi adatti per organizzare i dati. Le strutture dati scelte per il modello logico devono rendere facili le operazioni di manipolazione e veloci le attività di ritrovamento dei dati richiesti.
Il modello logico si ricava applicando alcune semplici regole di derivazione.

1. ogni entità diventa un archivio;
2. ogni attributo di un'entità diventa un campo dell'archivio, cioè il nome di un campo della struttura del record;
3. ogni campo del record eredita le caratteristiche dell'attributo dell'entità da cui deriva;
4. l'identificatore univoco di un'entità diventa la chiave primaria nel record dell'archivio derivato;
5. l'associazione uno a uno diventa un archivio unico, il cui tracciato record contiene i campi corrispondenti agli attributi della prima e della seconda entità;
6. l'identificatore univoco dell'entità di partenza nell'associazione uno a molti diventa **chiave esterna (foreign key)** dell'entità di arrivo associata, cioè l'attributo che è identificatore univoco diventa un campo nel record del secondo archivio;
7. l'associazione con grado molti a molti diventa un nuovo archivio (in aggiunta agli archivi derivati dalle due entità), il cui record contiene gli identificatori univoci delle due entità e gli eventuali attributi dell'associazione.

Gli esempi seguenti mostrano come si possa passare dallo schema ER alla struttura degli archivi nel livello logico e come si possano rappresentare le associazione tra entità nel modello relazionale.

La struttura dei dati viene rappresentata con il nome assegnato all'archivio, convertendo al plurale il nome dell'entità , seguito , tra parentesi tonde, dall'elenco dei campi corrispondenti agli attributi , separati da virgola.

Gli attributi sottolineati indicano la chiave dell'archivio.

Si consideri l'associazione tra l'entità CodiceFiscale e l'entità Cittadino: è un'associazione uno a uno in quanto ad ogni cittadino è assegnato uno e un solo codice fiscale.

Lo schema Entity/Relationship è il seguente:

A cui corrisponde l'archivio avente gli attributi dell'una e dell'altra entità:

AnagrafiTributarie (CodiceFiscale, Cognome, Nome, DataNascita)

Il campo CodiceFiscale diventa chiave primaria.

archivio	campo	chiave	formato	dimensione
AnagrafiTributarie	CodiceFiscale	Primaria	carattere	16
	Cognome		carattere	30
	Nome		carattere	25
	DataNascita		data/ora	8

In generale dall'associazione uno a uno del modello concettuale viene derivato un unico archivio che contiene gli attributi della prima e della seconda entità.

Nel seguente schema Entity/Relationship, l'associazione tra l'entità Contratto e l'entità Dipendente è una relazione uno a molti, in quanto ogni contratto può essere sottoscritto da uno o più dipendenti, e ogni dipendente deve essere regolato da un solo contratto di lavoro.

Viene aggiunta una chiave esterna (**foreign key)** nell'archivio Dipendenti , costituita dall'insieme degli attributi che formano la chiave primaria dell'archivio Contratti ed è rappresentata dal carattere evidenziato in corsivo.

Contratti (Codice, Descrizione, StipendioBase, DataScadenza)

Dipendenti(Matricola , Cognome, Nome, Indirizzo, Qualifica, ***CodiceContratto***)

archivio	campo	chiave	formato	dimensione
Contratti	Codice	primaria	numerico	3
	Descrizione		carattere	30
	StipendioBase		numerico	10
	DataScadenza		data/ora	8
Dipendenti	Matricola	primaria	carattere	7
	Cognome		carattere	30
	Nome		carattere	25
	Indirizzo		carattere	30
	Qualifica		carattere	10
	CodiceContratto	esterna	numerico	3

Agli attributi dell'entità Dipendente (a molti) è stata aggiunta come chiave esterna CodiceContratto, cioè la chiave dell'entità Contratto (uno).

In generale dal modello concettuale vengono derivate le tabelle che rappresentano le entità e l'associazione uno a molti viene tradotta aggiungendo agli attributi dell'entità "a molti" , la chiave dell'entità "a uno".

Si considerino l'entità Docente e l'entità Classe per una scuola: tra le due entità si può stabilire un'associazione molti a molti, in quanto ogni docente può insegnare in una o più classi, e ogni classe deve avere uno o più docenti.

Accanto agli archivi Docenti e Classi, viene creato un nuovo archivio, che chiameremo Insegna, il cui record contiene gli attributi chiave dei due archivi considerati.

Se vengono considerati anche gli attributi dell'associazione, questo nuovo archivio di legame contiene, oltre alle chiavi delle due entità anche gli attributi assegnati all'associazione: il numero delle ore di insegnamento del docente in una classe non è attributo né di **Docente**, né di **Classe**, ma dell'associazione; esso diventa un campo nel nuovo archivio ***Insegna***

Docenti (Codice, Nome, Qualifica, Materia)

Classi (Sigla, NumeroAlunni, Aula)

Insegna (***CodiceDocente, SiglaClasse***, NumeroOre)

archivio	campo	chiave	formato	dimensione
Docenti	Codice	primaria	carattere	5
	Nome		carattere	40
	Qualifica		carattere	7
	Materia		carattere	3
Classi	Sigla	primaria	carattere	3
	NumeroAlunni		numerico	2
	Aula		numerico	3
Insegna	CodiceDocente	esterna	carattere	5
	SiglaClasse	esterna	carattere	3
	NumeroOre		numerico	2

In generale quindi vengono derivati gli archivi corrispondenti alle entità e l'associazione molti a molti viene tradotta con un terzo archivio contenente le chiavi delle due entità e gli eventuali attributi dell'associazione.

Le basi di dati

Con il termine **basi di dati** (in inglese **database**) si indicano in informatica gli archivi di dati, organizzati in modo integrato attraverso tecniche di modellazione dei dati e gestiti sulle memorie di massa dei computer attraverso appositi software, con l'obiettivo di raggiungere una grande efficienza nel trattamento e nel ritrovamento dei dati, superando anche i limiti presenti nelle organizzazioni tradizionali degli archivi.

A grandi linee, possiamo dire che il database è una collezione di archivi di dati ben organizzati e ben strutturati, in modo che possano costituire una base di lavoro per utenti diversi con programmi diversi.

Per esempio i dati relativi agli articoli del magazzino di un'azienda possono essere utilizzati dal programma che stampa le fatture, oppure dal programma che stampa il listino prezzi.

Quando si parla di efficienza e di produttività di un'organizzazione di archivi si intende naturalmente la possibilità di ritrovare facilmente le informazioni desiderate, anche attraverso criteri di ricerca diversi, in termini di velocità nell'elaborazione, di sicurezza dei dati e integrità delle registrazioni, specialmente quando la gestione si riferisce a una mole di dati rilevante.

Infatti deve essere garantita la **consistenza** degli archivi, cioè i dati in essi contenuti devono essere significativi ed essere effettivamente utilizzabili nelle applicazioni dell'azienda.

I dati devono quindi essere protetti per impedire perdite accidentali dovute a cadute del sistema, guasti hardware o interventi dannosi da parte di utenti e di programmi; la protezione deve riguardare anche gli interventi dolosi sui dati dovuti ad accessi non autorizzati con operazioni di lettura, modifica o cancellazione.

In sostanza **sicurezza** significa impedire che il database venga danneggiato da interventi accidentali o non autorizzati; **integrità** significa garantire che le operazioni effettuate sul database da utenti autorizzati non provochino una perdita di consistenza ai dati.

I prodotti software per la gestione di database vengono indicati con il termine **DBMS** (DataBase Management System) .

Occorre anche sottolineare l'importanza che l'uso dei database assume nella gestione degli archivi di dati, favorendo l'utente del computer nel suo modo di vedere i dati, e liberandolo dagli aspetti riguardanti la collocazione fisica delle registrazioni sui supporti magnetici degli archivi.

I limiti dell'organizzazione convenzionale degli archivi

Le tecniche di gestione delle basi di dati nascono per superare i problemi e i limiti insiti nelle tradizionali organizzazioni degli archivi in modo non integrato. Gli archivi non integrati venivano utilizzati in passato per piccole applicazioni, tale tecnica consiste nel creare singoli file per creare archivi di dati gestiti da applicazioni singole. E' intuitivo capire che questo approccio può dar luogo a diversi problemi. Per esempio: se una Banca utilizza un file per conservare i dati dei propri clienti e un file per conservare i dati dei relativi conti correnti, bisogna preoccuparsi di mantenere «sincronizzati» i dati tra i due archivi. Se la Banca permette l'aggiornamento degli archivi da parte di più filiali, i problemi aumentano.

Pertanto, i database nascono proprio per superare i limiti e gli inevitabili problemi che si potrebbero incontrare usando gli archivi tradizionali. Tra i classici problemi vanno ricordati la *ridondanza* dei dati e l'*inconsistenza* degli archivi. Il primo caso si verifica quando in diversi archivi si trovano memorizzati gli stessi tipi di dati (il problema è quello di dover fare gli aggiornamenti in tutti gli archivi interessati). Il secondo caso si verifica come conseguenza della ridondanza, quando questi aggiornamenti non vengono fatti in tutti gli archivi in cui si dovrebbe, e nascono così incongruenze tra dati nuovi e dati vecchi.

Gli archivi integrati ed i database distribuiti

Il concetto di integrazione degli archivi e di base di dati forniscono l'idea di dati accentrati. In realtà gli archivi integrati che costituiscono la base di dati aziendale possono risiedere su un unico computer oppure possono essere distribuiti sulle memorie di massa di computer diversi facenti parte di una rete aziendale, i cui nodi possono essere anche fisicamente lontani: in questo caso si parla di **database distribuiti.** Gli utenti della base di dati elaborano in modo locale gli archivi che hanno a disposizione nel proprio sistema e nello stesso tempo accedono in modo remoto a sistemi centrali attraverso le linee di comunicazione .

Il vantaggio offerto consiste nella possibilità di gestire archivi di dimensioni limitate facilitando il lavoro di manutenzione e di controllo sulla sicurezza, garantendo comunque la disponibilità dei dati aggiornati a tutti gli utenti del sistema informativo aziendale, qualunque sia la loro posizione geografica o il computer da essi usato per l'attività di elaborazione.

L'integrazione degli archivi è realizzata quindi a livello logico, nella visione che di essi hanno gli utenti finali, anche se i dati fisicamente possono risiedere su sistemi distanti.

I modelli per il database

Il database è un modello della realtà considerata: i contenuti della base di dati rappresentano gli stati in cui si trova la realtà da modellare. I cambiamenti che vengono apportati alla base di dati rappresentano gli eventi che avvengono nell'ambiente in cui opera l'azienda. L'uso dei dati organizzati in un database presuppone un attento lavoro di progettazione iniziale, che viene fatto con riferimento ai dati che si vogliono memorizzare e successivamente elaborare.

Il progetto è indipendente dal computer, dai supporti fisici destinati a contenere le informazioni e dalle caratteristiche del DBMS. Questo è un elemento di novità rispetto all'organizzazione convenzionale degli archivi per la quale il lavoro e la programmazione erano molto legate alle caratteristiche delle risorse dell'hardware e del linguaggio a disposizione.

Questo modo di operare consente alla base di dati di evolvere nel tempo insieme alla realtà aziendale per la quale è stata progettata, con il crescere delle esigenze degli utenti e della necessità di informazioni.

Modelli di database

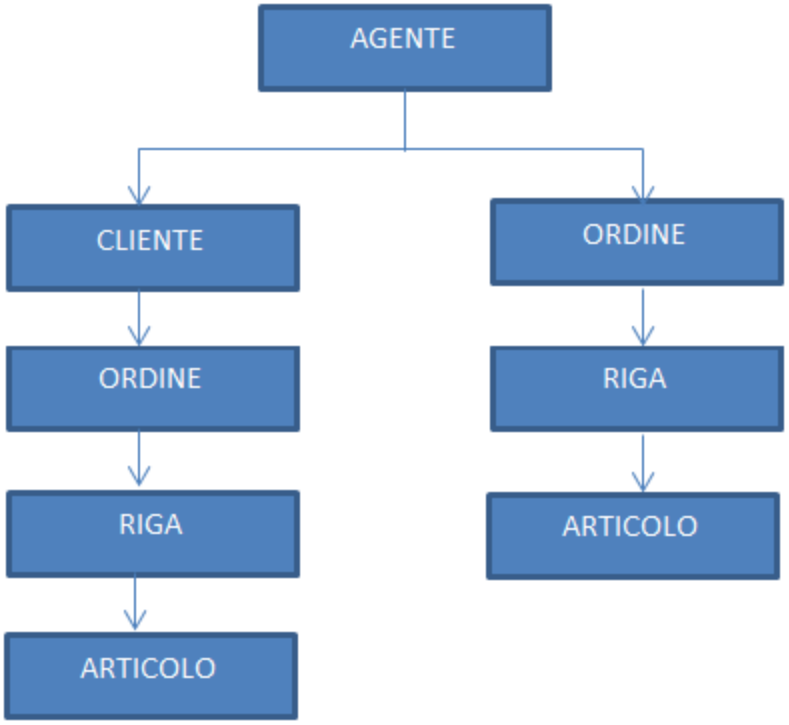
A partire dallo schema concettuale E/R, un database può essere progettato e realizzato passando al modello logico, cioè alle strutture di dati che consentono di organizzare i dati per consentire le operazioni di manipolazione e di interrogazione.

La soluzione più semplice consiste nel costruire un database con una struttura di dati formata da un unico file detta **flat file** (tipo excel). Essa però è adatta solamente per basi di dati molto semplici.

Nello sviluppo della teoria dei database, dal 1960 in poi, sono emersi principalmente tre tipi diversi di modelli per le basi di dati:

Modello Gerarchico

E' particolarmente adatto per rappresentare situazioni nelle quali è possibile fornire all'insieme dei dati una struttura nella quale ci sono entità che stanno in alto ed entità che stanno in basso, secondo uno schema ad albero, nel quale i nodi rappresentano le entità e gli archi rappresentano le associazioni. Nella pratica l'entità è un file, l'istanza è un record e gli attributi sono i campi del record. E' particolarmente adatto a rappresentare le associazioni 1:N. Presenta dei limiti soprattutto nella rigidità della struttura di dati creata, che talvolta non riesce ad evitare ridondanze dei dati.

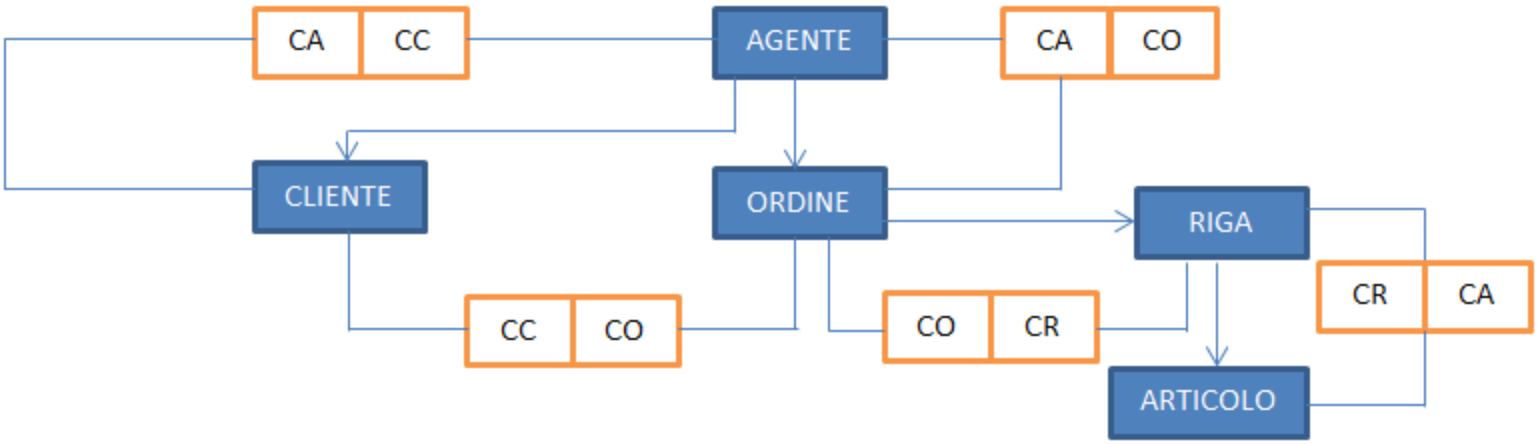


Nonostante questi limini , il modello gerarchico ha avuto in passato una larga diffusione, per il grande numero di implementazioni su modelli diversi di computer, e anche perché molte situazioni presentano in modo naturale una struttura di dati di tipo gerarchico.

Modello Reticolare

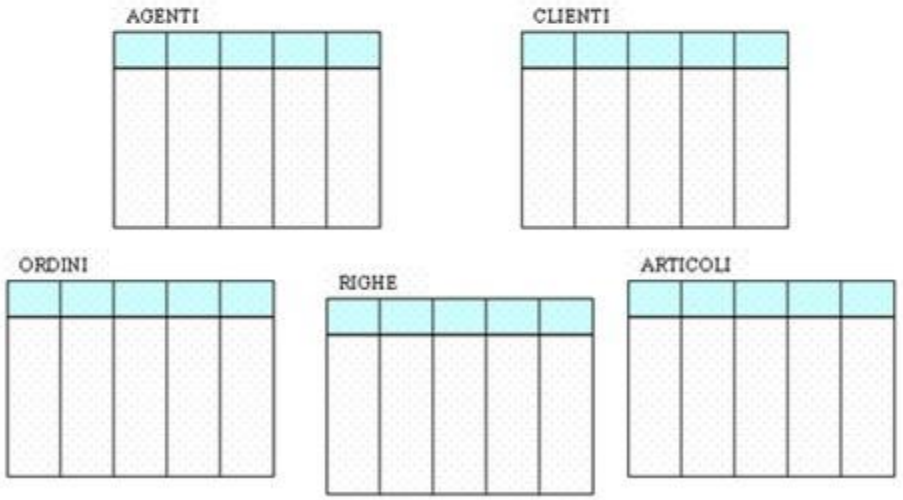
In questo modello le entità rappresentano i nodi e le associazioni rappresentano gli archi di uno schema a grafo orientato: si tratta cioè di una estensione del modello gerarchico. Non esiste quindi una gerarchia predefinita tra le entità, un record figlio può avere un numero qualsiasi di padri: in questo modo vengono evitate situazioni di ripetizione di dati uguali.

Risulta però più difficile l'implementazione e la costruzione del software applicativo. Ha avuto una discreta diffusione negli anni '70 soprattutto nei sistemi di grandi dimensioni.



Modello Relazionale

Rappresenta il database come un insieme di tabelle. Esso viene considerato attualmente il modello più semplice ed efficace, perché è più vicino al modo consueto di pensare i dati, e si adatta in modo naturale alla classificazione e alla strutturazione dei dati.



Il modello relazionale nasce nel 1970, proposto da Edward F. Codd, ricercatore dell'IBM come idea di un modello logico molto semplice e nello stesso tempo in grado di superare i limiti degli altri modelli utilizzati. Entra a far parte di sistemi commerciali a partire dal 1978. Esso si basa su alcuni concetti fondamentali tipicamente matematici e assegna grande importanza all'uso rigoroso del linguaggio matematico, con due obiettivi importanti:

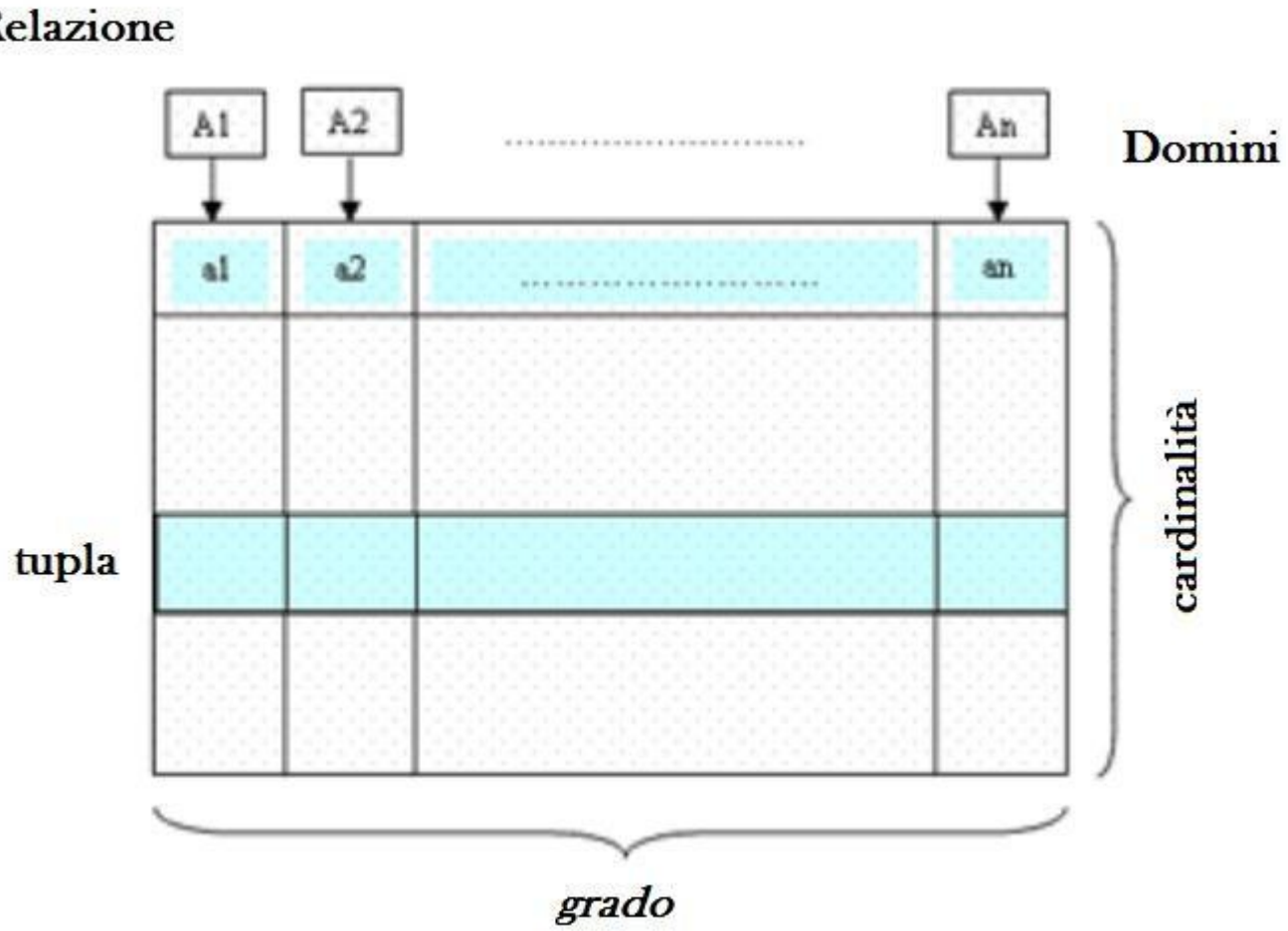
1. utilizzare un linguaggio conosciuto a livello universale, ossia il linguaggio matematico
2. eliminare i problemi di ambiguità nella terminologia e nella simbologia

Con l'introduzione del livello relazionale, ben presto i modelli gerarchico e reticolare sono diventati obsoleti per il mercato e per la ricerca. Questo perché le operazioni sui database gerarchici sono complesse, agiscono sui singoli record e non su gruppi di record. I modelli non relazionali sono fondamentalmente basati sulla programmazione di applicazioni, l'utente deve specificare i percorsi per ritrovare i dati e la velocità nel ritrovare le informazioni dipende dal software di gestione; nel modello relazionale invece i percorsi per le interrogazioni sono a carico del sistema.

Il modello relazionale è più intuitivo e più espressivo per la strutturazione dei dati, rispetto agli altri modelli però è relativamente più lento, nella ricerca e occupa più spazio su memoria di massa rispetto ai database creati e gestiti da DBMS basati sugli altri modelli.

Vediamo in dettaglio il Modello Relazionale.

Esso si chiama così perché è fondato sul concetto matematico di **relazione** tra insiemi di oggetti.



Dal punto di vista matematico, dati n insiemi A_1, A_2,……, An, si dice **relazione** un sottoinsieme dell'insieme di tutte le n-uple a_1,a_2,……,a_n che si possono costruire prendendo nell'ordine un elemento a_1 dal primo insieme A_1, a_2 dal secondo insieme A_2 e così via.

Il numero n si chiama **grado** della relazione, gli insiemi Ai si chiamano **domini** della relazione, e il numero delle n-uple (indicate con il termine **tuple**) si chiamano **cardinalità** della relazione.

La relazione viene comunemente rappresentata con una tabella, avente tante colonne quanti sono i domini (grado della relazione) e tante righe quante sono le n-uple (cardinalità della relazione). I nomi dei domini sono i nomi delle colonne, i valori che compaiono in una colonna sono omogenei tra loro, cioè appartengono a uno stesso dominio. La relazione è quindi una collezione di n-uple, ciascuna delle quali contiene i valori di un numero prefissato di colonne. La relazione rappresenta un'entità, ogni n-upla rappresenta un'istanza dell'entità, le colonne contengono gli attributi dell'entità, il dominio è l'insieme dei valori che possono essere assunti da un attributo.
La **chiave** della relazione è un attributo o una combinazione di attributi che identificano univocamente le n-uple all'interno della relazione, cioè ogni riga della tabella possiede valori diversi per l'attributo (o gli attributi) chiave.
Per esempio, la relazione Computer può essere rappresentata con un a tabella del tipo:

Modello	Produttore	Distributore	Processore	Prezzo

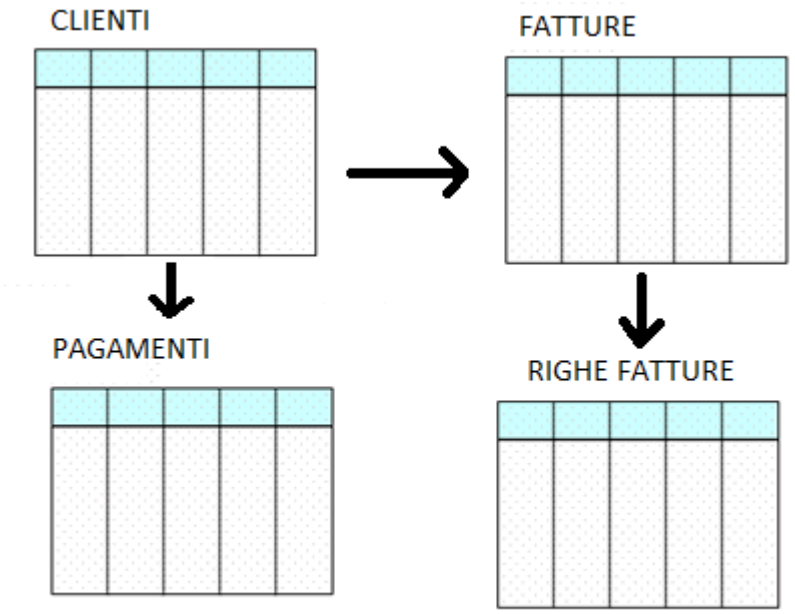
La chiave è rappresentata dall'attributo Modello.

Secondo una notazione orami consolidata nella rappresentazione delle tabelle si usa una scrittura del tipo:

Computer (Modello, Produttore, Distributore, Processore, Prezzo)

Indicando tra parentesi, dopo il nome della relazione, i nomi degli attributi e sottolineando l'attributo chiave.

Il modello relazionale di un database è un insieme di tabelle sulle quali si possono effettuare operazioni, e tra le quali possono essere stabilite associazioni, come vedremo di seguito.



Requisiti del Modello Relazionale

Essi sono:

1. tutte le righe della tabella contengono lo **stesso numero di colonne**, corrispondente agli attributi;
2. gli attributi rappresentano **informazioni elementari (o atomiche),** non scomponibili ulteriormente, cioè non ci sono campi di gruppo che contengono per ogni riga un insieme di valori anziché un solo valore;
3. i valori assunti da un campo appartengono al **dominio** dei valori possibili per quel campo, e quindi sono **valori omogenei** tra loro, cioè sono dello stesso tipo;
4. in una relazione, ogni riga è **diversa** da tutte le altre, cioè non ci possono essere due righe con gli stessi valori dei campi: questo significa che esiste un attributo o una combinazione di più attributi che identificano univocamente la n-upla, e che assumono perciò la funzione di **chiave primaria** della relazione;
5. le n-uple compaiono nella tabella secondo **un ordine non prefissato**, cioè non è rilevante il criterio con il quale le righe sono sistemate nella tabella.

Il modello relazionale fissa una regola di integrità sui dati, detta **integrità sull'entità (entity integrity)** secondo la quale nessuna componente chiave primaria può avere valore nullo.

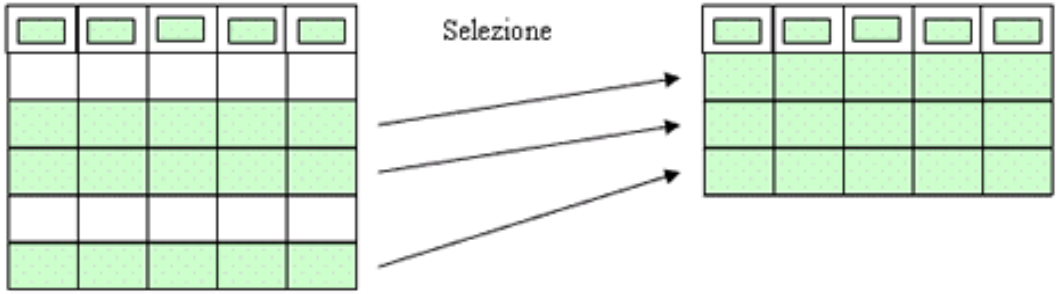
Le **regole di derivazione** applicabili al modello E/R sono le stesse che abbiamo visto sopra per il modello logico in generale.

Le operazioni relazionali

In generale gli **operatori relazionali** agiscono su una relazione per ottenere una nuova relazione. In sostanza le operazioni relazionali consentono di effettuare le **interrogazioni** sulla base di dati per ottenere le informazioni desiderate estraendo da una tabella una sottotabella, oppure combinando tra loro due o più tabelle e generando così nuove relazioni.
Le principali operazioni sono:

1. La **selezione (o select)** che genera una nuova relazione è costituita solo dalle n-uple della relazione di partenza che soddisfano a una determinata condizione.

   Grado: lo stesso della relazione di partenza
   Cardinalità: minore o uguale a quella di partenza



Per esempio, se si vuole l'elenco dei clienti della provincia di Milano, si effettua sulla relazione Clienti una selezione per Provincia="MI", estraendo dalla tabella tutte le righe che hanno quel valore per l'attributo Provincia, ottenendo così una nuova tabella , che costituisce la relazione generata a partire da quella iniziale.

Agenti

CodiceAgente	Nome Agente	Indir Agente	Codice Zona

Clienti

CodiceCliente	Ragione Sociale	CodiceAttività	Partita IVA	Indir Cliente	Provincia	***CodiceAgente***

Utilizzando un linguaggio di progetto (o di pseudocodifica), cioè con parole del linguaggio naturale, l'operazione di selezione può essere descritta in questo modo:

Selezione di Clienti per Provincia="MI"

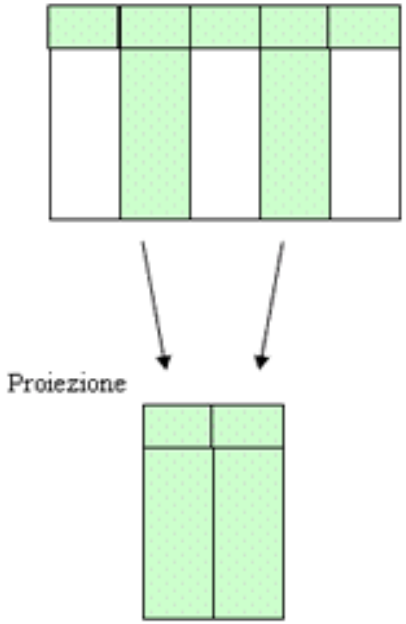
2. La **proiezione (o project)** che genera una nuova relazione estraendo dalla tabella iniziale due o più colonne corrispondenti agli attributi prefissati. La tabella ottenuta potrebbe contenere alcune righe uguali: in questo caso occorre richiedere che ne venga conservata una sola, perché il modello relazionale tra i suoi requisiti fondamentali, come già detto, non consente righe uguali tra loro.

Grado: minore o uguale al grado della relazione di partenza
Cardinalità: di solito uguale a quella di partenza tranne quando vengono ridotte a una le righe uguale con la unique.

Per esempio, se si vuole l'elenco dei codici di attività dei clienti con i relativi codici degli agenti, occorre applicare alla relazione **Clienti** l'operazione di **proiezione** secondo gli attributi **CodiceAttività** e **CodiceAgente**: dopo aver eventualmente ridotto ad una le righe uguali, si ottiene una nuova relazione che contiene le informazioni desiderate.

In pseudocodifica si scrive:

Proiezione di Clienti su CodiceAttività , CodiceAgente



3. La **congiunzione (o join)** serve a combinare due relazioni aventi uno o più attributi in comune, generando una nuova relazione che contiene le righe della prima e della seconda tabella, che possono essere combinate secondo i valori uguali dell'attributo comune.

Grado: se i gradi (cioè il numero delle colonne) delle relazioni di partenza sono rispettivamente N_1 e N_2, il grado della relazione generata è uguale a N_1+N_2 - 1, perché l'attributo comune compare una sola volta.

Cardinalità: non è prevedibile in quanto si ottengono solo le righe che possono essere combinate attraverso valori presenti in entrambe.

Questo tipo di congiunzione viene detta join naturale (**natural join**)

Per esempio, l'elenco dei clienti e dei relativi agenti si ottiene applicando l'operazione di congiunzione tra la relazione Clienti e la relazione Agenti secondo l'attributo comune CodiceAgente.

Questa operazione si rappresenta in pseudocodifica nel seguente modo:

Congiunzione di Clienti su CodiceAgente e di Agenti su CodiceAgente

La congiunzione viene realizzata facendo corrispondere valori uguali per attributi comuni nelle due tabelle: per questo motivo si chiama **equi-join**. Possiamo avere altri tipi di congiunzione.

Riportiamo di seguito le join possibili:

1. **natural join** congiunzione vista in precedenza
2. **equi-join** quando si fanno corrispondere valori uguali per attributi comuni nelle due tabelle (esempio visto nella pagina precedente: Clienti.CodiceAgente su Agenti.CodiceCliente).
3. **Join esterno (outer join)** è una congiunzione che restituisce le righe dell'una e dell'altra tabella anche se non sono presenti valori uguali per l'attributo comune; di tutte queste vengono combinate solo le righe per le quali il valore dell'attributo comune nella prima tabella trova un valore uguale nella colonna dell'attributo comune nella seconda tabella.

   Queste si possono ulteriormente dividere in:

   left join = quando elenca comunque tutte le righe della prima tabella congiungendo tra le righe della seconda solo quelle per le quali si trovano valori corrispondenti per l'attributo comune.

   right join = restituisce comunque tutte le righe della seconda tabella e di queste congiunge con le righe della prima tabella solo quelle per le quali si possono trovare valori corrispondenti per l'attributo comune.

   self-join = è una congiunzione con la quale vengono combinate righe di una tabella con le righe della stessa tabella quando sono presenti valori corrispondenti per attributi, cioè due attributi con lo stesso dominio.

La Normalizzazione delle relazioni

Nella definizione della struttura di una relazione occorre evitare la ripetizione e la ridondanza dei dati per non creare problemi nella fase di trattamento della tabella con operazioni di modifica o cancellazione di righe. Quindi a partire da una data relazione, vengono create altre relazioni, secondo un processo che viene indicato con il termine di **normalizzazione**: la definizione di relazioni normalizzate avviene a livelli crescenti di normalizzazione, a cui corrispondono diverse **forme normali** della relazione.

La normalizzazione è un processo formalizzato con il quale le tabelle vengono trasformate in modo tale che ogni tabella corrisponda a un singolo oggetto della realtà rappresentata con il modello di database: le regole della normalizzazione sono definite per evitare l'inconsistenza dei dati e le anomalie nelle operazioni di aggiornamento.

In sostanza la normalizzazione consente di creare tabelle ben definite, che facilitano le operazioni di aggiunta, modifica e cancellazione delle informazioni , e che rendono possibili i cambiamenti nella struttura del modello con l'evolvere delle esigenze aziendali e degli utenti del database.

In ogni caso deve essere garantito che la scomposizione di una tabella in altre tabelle di forma normale superiore non provochi perdita di dati.

Nel seguito della trattazione verranno utilizzati alcuni termini di cui si forniscono le definizioni:

-la **chiave o chiave primaria** è l'insieme di uno o più attributi che identificano in modo univoco una n-upla (riga della tabella);

- l'**attributo non-chiave** è un campo che non fa parte della chiave primaria;

- la **dipendenza funzionale** tra attributi si ha quando il valore di un attributo A1 determina un singolo valore dell'attributo A2 e si indica con A1 -> A2

-la **dipendenza transitiva** si ha quando un attributo A2 dipende da A1 e l'attributo A3 dipende da A2; allora A3 dipende transitivamente da A1

A1 -> A2 e A2 -> A3 allora A1 -> A3 in modo transitivo

Vengono esaminate ora le principali forme normali nel modello relazionale.

Prima forma normale

Una relazione è in **prima forma normale (1FN)** quando rispetta i requisiti fondamentali del modello relazionale che sono:

1. **tutte le righe** della tabella contengono lo **stesso numero di colonne**;
2. **gli attributi** rappresentano **informazioni elementari**;
3. **i valori** che compaiono in una **colonna** sono dello **stesso tipo**, cioè appartengono allo **stesso dominio**;
4. **ogni riga** è **diversa da tutte le altre**, cioè non ci possono essere due righe con gli stessi valori nelle colonne;
5. l'**ordine** con il quale le **righe** compaiono nella tabella è **irrilevante**.

In particolare **gli attributi devono essere informazioni non ulteriormente scomponibili**, cioè non devono avere sottoattributi, né essere gruppi di attributi ripetuti.

Per esempio , nella relazione

Dipendenti (Matricola, Nome, Indirizzo, FamiliariACarico)

L'attributo FamiliariACarico non è elementare, in quanto è costituito da un gruppo di attributi ripetuti dello stesso tipo (i nomi dei familiari). **La relazione non è in prima forma normale.**

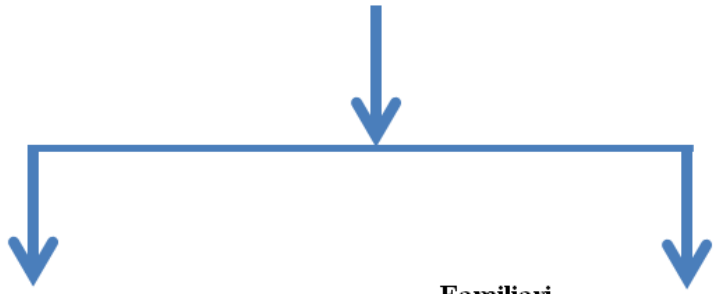
La relazione può essere opportunamente convertita in due tabelle.

Dipendenti (Matricola , Nome, Indirizzo)

Familiari (CodiceFam, NomeFam, MatricolaDIP)

Dipendenti

Matricola	Nome	Indirizzo	Familiari a Carico



Dipendenti

Matricola	Nome	Indirizzo

Familiari

CodiceFam	NomeFam	MatricolaDip

In questo modo vengono individuati due oggetti distinti , Dipendente e Familiare , fornendo una rappresentazione più precisa della realtà: inoltre risulta più facile aggiungere eventuali nuovi attributi (per esempio, età , sesso, grado di parentela alla relazione Familiari).

Anche l'inserimento di un nuovo figlio a carico del dipendente diventa semplice, in quanto viene aggiunta una riga alla tabella Familiari, senza modificare la struttura delle relazioni; inoltre possiamo comunque ritrovare le informazioni sul capofamiglia attraverso un'operazione di congiunzione tra la relazione Dipendenti e la relazione Familiari sull'attributo comune Matricola del dipendente.

Le forme normali superiori alla prima vengono introdotte per eliminare problemi durante le operazioni di aggiornamento o di cancellazione, evitando l'inconsistenza o la perdita indesiderata di dati.

Seconda forma normale

Una relazione è in **seconda forma normale (2FN)** quando è in prima forma normale e tutti i suoi attributi non-chiave dipendono dall'intera chiave, cioè non possiede attributi che dipendono soltanto da una parte della chiave. La seconda forma normale elimina la dipendenza parziale degli attributi dalla chiave e riguarda il caso di relazioni con chiavi composte, cioè formate da più attributi.

Utilizzando la simbologia descritta nelle definizioni precedenti di dipendenza, il passaggio alla seconda forma normale si schematizza nel seguente modo:

Si consideri per esempio il problema dell'inventario delle merci che si trovano in alcuni magazzini dislocati in località diverse. Le informazioni essenziali possono essere rappresentate con la seguente relazione:

Merci (Codice, Magazzino, Quantità, LocalitàMagazzino)

Merci

Codice	Magazzino	Quantità	LocalitàMagazzino

La chiave è composta, formata dagli attributi Codice e Magazzino, in quanto il solo codice non identifica la merce, che può essere presente in magazzini diversi.

L'indirizzo del magazzino riguarda soltanto l'attributo Magazzino : nella relazione è presente un attributo LocalitàMagazzino che dipende soltanto da una parte della chiave.

La relazione non è in seconda forma normale e può provocare problemi di questo genere:

- la località del magazzino è ripetuta per tutte le righe della tabella che si riferiscono a prodotti presenti in quel magazzino;
- se la località del magazzino cambia, ogni riga contenente merci presenti in quel magazzino deve essere aggiornata;
- la ridondanza può provocare l'inconsistenza delle informazioni perché la località potrebbe essere scritta in modo differente in righe diverse per lo stesso magazzino , oppure perché potrebbe accadere che alcuni record vengono aggiornati ed altri no;
- se in un certo periodo non ci fossero merci presenti in un magazzino, non avremmo alcuna informazione sulla località del magazzino; la cancellazione di righe può determinare una perdita complessiva di informazioni nella base di dati.

La soluzione consiste nel costruire nuove relazioni, togliendo dalla relazione di partenza gli attributi che dipendono solo parzialmente dalla chiave primaria.

Merci(Codice, Magazzino, Quantità)
Depositi (Magazzino, LocalitàMagazzino)

Merci

Codice	Magazzino	Quantità

Depositi

Magazzino	LocalitàMagazzino

La relazione, nella quale non ci sono attributi non-chiave che dipendono funzionalmente solo da una parte della chiave, si dice in seconda forma normale **(2FN)**

Esempio di relazione

R1(A1,A2, A3, A4, A5)

Con (A1,A2) ->A3

A1->A4

A2->A5

Non è in 2FN, e può essere trasformata nelle seguenti relazioni in 2FN

R21(A1,A2, A3)

R22(A1, A4)

R23(A2, A5)

Il processo attraverso il quale la struttura della relazione è stata trasformata in tre nuove relazioni rappresenta il significato del termine **normalizzazione**.

Si deve osservare che la normalizzazione diminuisce la ridondanza dei dati e la possibilità di inconsistenza, ma rende più complesse le operazioni di ritrovamento dei dati perché le query devono abbracciare più tabelle.

Terza forma normale

Una relazione è in **terza forma normale (3FN)** quando è in seconda forma normale e tutti gli attributi non-chiave dipendono direttamente dalla chiave, cioè non possiede attributi non-chiave che dipendono da altri attributi non-chiave. La terza forma normale elimina la dipendenza transitiva degli attributi dalla chiave.
Per esempio si consideri la gestione anagrafica di un'associazione di studenti di scuole diverse. Le informazioni più importanti sono rappresentate con la seguente relazione:

Studenti (Nome, Scuole , TelefonoScuola)

Studenti

Nome	Scuola	Telefono Scuola

Il nome è l'attributo chiave e il telefono della scuola dipende dalla scuola, oltre che essere un'informazione che riguarda lo studente. Nella relazione è presente un attributo non-chiave (TelefonoScuola) che dipende da un altro attributo non-chiave (Scuola) . La relazione non è in terza forma normale in quanto l'attributo TelefonoScuola dipende transitivamente dalla chiave (Nome).
In modo analogo a quanto visto per la seconda forma normale, anche in questo caso si possono avere anomalie nell'aggiornamento e inconsistenza dei dati;

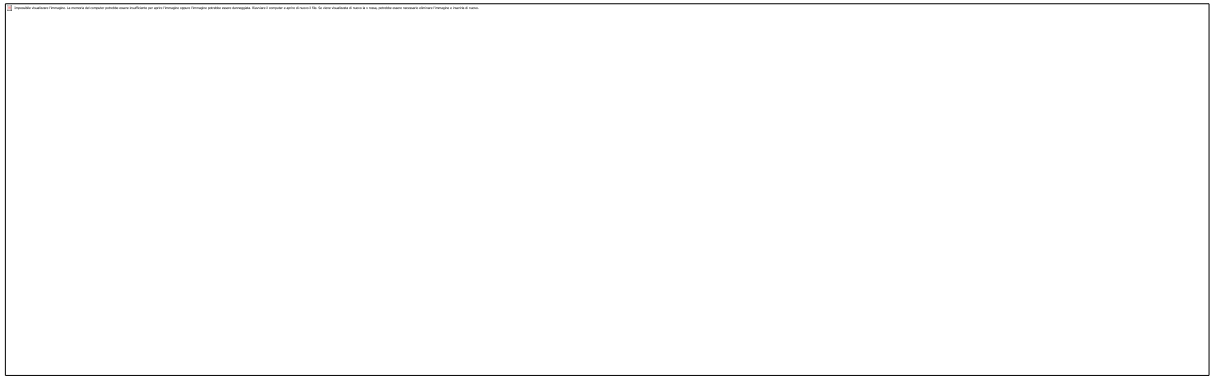
- il telefono della scuola è ripetuto per ogni studente appartenente a quella scuola;
- se il telefono della scuola cambia, occorre modificare tutte le righe contenenti studenti di quella scuola;

- la ridondanza può provocare inconsistenza, in quanto ci possono essere telefoni differenti in righe diverse per la stessa scuola, nel caso in cui siano scritti in modo diverso oppure l'aggiornamento non sia stato fatto su tutte le righe.
- Se una scuola non ha nessun studente appartenente all'associazione oppure gli studenti iscritti di una scuola escono dall'associazione, non possiamo avere alcuna informazione sul telefono della scuola, con una perdita complessiva di informazione.

La normalizzazione in 3FN si ottiene scomponendo la relazione di partenza in due nuove relazioni, nelle quali tutti gli attributi dipendono direttamente dalla chiave, togliendo gli attributi non-chiave che dipendono da un altro attributo non-chiave.

Studenti (Nome, Scuola)

Istituti (Scuola, TelefonoScuola)



La relazione, nella quale non ci sono attributi non-chiave che dipendono funzionalmente da un altro attributo non-chiave (cioè che dipendono in modo transitivo dalla chiave) , si dice in terza forma normale (3FN).

Utilizzando la simbologia vista in precedenza, il passaggio alla terza forma normale si schematizza nel seguente modo:

R2(<u>A1,</u>A2,A3,A4)

Con A2->A4

Non è in 3FN e può essere trasformata nelle seguenti relazioni in 3FN

R31(<u>A1</u>,A2,A3)

R32(<u>A2</u>,A4)

Il significato delle forme normali viene di seguito riassunto:

-**Prima forma normale**: una relazione si dice in prima forma normale quando rispetta i requisiti fondamentali del modello relazionale, in particolare ogni attributo è elementare, non ci sono righe uguali e non ci sono attributi ripetitivi.
-**Seconda forma normale**: quando è in prima forma normale e non ci sono attributi non-chiave che dipendono parzialmente dalla chiave.
-**Terza forma normale**: quando è in seconda forma normale e non ci sono attributi non-chiave che dipendono transitivamente dalla chiave

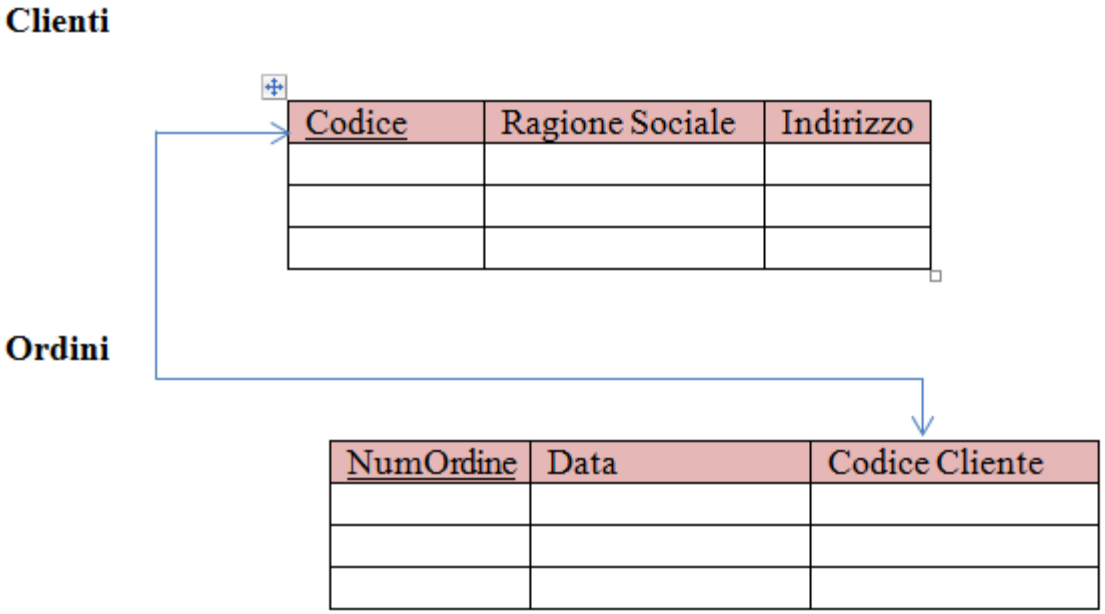
Il modello relazionale richiede come obbligatoria solo la prima forma normale. Tuttavia il passaggio alle forme normali superiori consente di distinguere e separare con precisione gli oggetti, senza perdita di informazioni, anche se viene generata una ridondanza di dati che però è costantemente sotto controllo.

Integrità Referenziale

L'**integrità referenziale** (referential integrity) è un insieme di regole del modello relazionale che garantiscono l'integrità dei dati quando si hanno relazioni associate tra loro attraverso la chiave esterna: queste regole servono per rendere valide le associazioni tra le tabelle e per eliminare gli errori di inserimento, cancellazione o modifica di dati collegati tra loro.
L'integrità referenziale viene rispettata quando per ogni valore non nullo della chiave esterna, esiste un valore corrispondente della chiave primaria nella tabella associata.

Per esempio in un database relazionale che contiene la tabella dei clienti e la tabella degli ordini, il codice del cliente della tabella Ordini è associato alla chiave della tabella Clienti.

Clienti (Codice, RagioneSociale, Indirizzo)
Ordini (NumOrdine, Data, CodiceCliente)



Applicare l'integrità referenziale al database significa garantire che un valore presente nella tabella Ordini per la chiave esterna CodiceCliente abbia un corrispondente valore di Codice in una delle righe della tabella Clienti. Inoltre non si deve consentire la cancellazione di un cliente dalla tabella Clienti se ci sono righe nella tabella Ordini che si riferiscono ad esso.

Quando viene applicata l'integrità referenziale, è necessario osservare le seguenti regole pratiche:

-Non è possibile immettere un valore nella chiave esterna della tabella associata, se tale valore non esiste tra le chiavi della tabella primaria.

-Non è possibile eliminare una n-upla dalla tabella primaria, se esistono righe legate ad essa attraverso la chiave esterna nella tabella correlata

- Non si può modificare, come è ovvio, il valore della chiave nella tabella primaria, se ad essa corrispondono righe nella tabella correlata.

La gestione del database

Il **DBMS** (Database Management System) è il software che consente di costruire e gestire una base di dati, realizzandola nella pratica su memoria di massa, a partire da un progetto e da uno schema dei dati definiti a livello concettuale e tradotto poi in un modello logico dei dati.

Il DBMS costituisce quindi un'interfaccia tra gli utenti di un database con le loro applicazioni e le risorse costituite dall'hardware e dagli archivi di dati presenti in un sistema di elaborazione.

Le funzioni che il DBMS, attraverso i suoi moduli software, è in grado di offrire agli utenti del database sono raggruppate nei seguenti aspetti principali:

1. **Implementazione del modello logico sul sistema di elaborazione**

- definizione dei dati e delle strutture dati derivare dallo schema logico (tipicamente le tabelle del modello relazionale), con produzione della documentazione sul modello;
- definizione dei sottoschemi (**viste**), cioè visioni del database legate alle particolari applicazioni dei singoli utenti e che consentono agli utenti di accedere ai dati che servono, ottenute attraverso proiezioni o congiunzioni; la vista si chiama anche **tabella virtuale**, rispetto alle tabelle del database, dette anche **tabelle primarie.** Le viste sono finestre dinamiche sulle tabelle del database, in quanto ogni modifica ai dati sulla tabella primaria si riflette sulla vista e viceversa;
- Organizzazione fisica dei dati sui supporti di memorizzazione, utilizzando le tecniche che ottimizzano l'occupazione della memoria di massa e i tempi di accesso alle registrazioni.

La gestione del database

2. Manipolazione e interrogazione sulla base di dati

•Inserimento dei dati nel database e trattamento dei dati già registrati con operazioni di modifica o cancellazione;

•Interfaccia tra i programmi degli utenti (scritti con i tradizionali linguaggi di programmazione) e la base di dati, utilizzando le funzionalità del DBMS per migliorare l'organizzazione dei dati e le prestazioni dei programmi nelle operazioni di ritrovamento dei dati;

•accesso ai dati contenuti nel database per le interrogazioni, attraverso interfacce o comandi semplici che facilitano soprattutto l'utente finale non specialista.

3. Controllo dell'integrità dei dati

- Integrità sulle entità;
- Integrità referenziale;
- Integrità definite dall'utente, cioè i vincoli che sono specifici per un particolare database, come conseguenza di politiche commerciali dell'azienda oppure di norme legislative e fiscali.

4. Sicurezza e protezione

- garanzia di sicurezza dei dati contro i danni causati da malfunzionamento di componenti hardware o software o da interventi dolosi;
- protezione dei dati da eventuali danneggiamenti per garantire l'integrità dei dati, offrendo anche la possibilità di attivare procedure di **recovery** in caso di perdita dei dati;
- Autorizzazione degli utenti che accedono alla base di dati e protezione dei dati dagli accessi non autorizzati;
- Controllo degli accessi in modo concorrente al database da parte di più utenti.

Il linguaggio SQL

Caratteristiche generali

Lo sviluppo e il raffinamento delle tecniche di gestione delle basi di dati hanno dato vita a linguaggi formati da comandi specifici, per consentire agli utenti un facile uso delle prestazioni del DBMS per basi di dati relazionali (detti RDBMS, cioè Relational DBMS).

Il linguaggio SQL (Structured Query Language) è un linguaggio non procedurale che è diventato uno standard tra i linguaggi per la gestione di database relazionali. Dopo una prima versione introdotta da IBM alla fine degli anni 1970, oggi il linguaggio SQL viene usato in tutti i prodotti DBMS come linguaggio di comandi per l'utente della base di dati (ricordiamo Oracle, Informix, SQLServer, Access, ecc.).

Secondo la terminologia del linguaggio SQL un database è costituito da tabelle, che rappresentano le relazioni; gli attributi sono le colonne della tabella e i record sono le righe della tabella.

Il linguaggio SQL consente di:

•Definire la struttura delle relazioni del database (funzioni **DDL** o Data Definition Language). Il DDL rappresenta lo strumento attraverso il quale l'utente, facendo riferimento al proprio schema logico, ordina al DBMS la creazione della struttura fisica del database e possiede inoltre specifici comandi per eliminare tabelle e viste già esistenti.

•Modificare i dati contenuti nel database, con le operazioni di inserimento, variazione e cancellazione (funzioni di **DML** o Data Manipulation Language).

•Gestire il controllo degli accessi e i permessi per gli utenti (funzioni di **DCL** o Data Control Language). IL DCL rappresenta lo strumento attraverso il quale l'utente fissa i vincoli di integrità, per stabilire le autorizzazioni agli accessi e i tipi di permessi consentiti agli utenti (inserimento di nuovi dati, sola lettura, modifica dei dati).

•Porre interrogazioni al database (**funzioni di Query Language**), che consente il ritrovamento dei dati che interessano, sulla base dei criteri di ricerca richiesti dall'utente.

Il linguaggio SQL fornisce inoltre opportuni comandi per definire in modo facile i tabulati di uscita dei risultati (**report**), per recuperare i dati quando accade un'interruzione del programma, un guasto nei supporti magnetici o un malfunzionamento del sistema, per definire le viste degli utenti sul database, per garantire la sicurezza dei dati nei confronti degli accessi di più utenti.

Il linguaggio SQL

Identificatori e tipi di dati

Gli identificatori (nomi di tabelle e di attributi) sono costituiti da sequenze di caratteri con lunghezza massima uguale a 18 caratteri : devono iniziare con una lettera e possono anche contenere il carattere _.

Quando è necessario identificare il nome di un attributo della tabella si deve usare la notazione

NomeTabella.NomeAttributo

(separati dal punto).

Nella dichiarazione della struttura di una tabella occorre specificare il tipo dei dati scelto per gli attributi.

I tipi standard del linguaggio SQL sono:

•*CHARACTER(n)*	*Stringa di lunghezza n*	*n da 1 a 15000*
•*DATE*	*Data nella forma MM/GG/AA*	
•*TIME*	*Ora nella forma HH: MM*	
•*INTEGER (p)*	*Numero intero con precisione p*	*p da 1 a 45*
•*SMALLINT*	*Numero intero con precisione 5*	*da -32768 a 32767*
•*INTEGER*	*Numero intero con precisione 10*	*da -2.147.483.648 a 2.147.483.647*
•*DECIMAL(p,s)*	*Numero decimale con precisione p e s cifre decimali*	*p da 1 a 45 ed s da 0 a p*
•*REAL*	*Numero reale con mantissa di precisione 7*	*valore 0 oppure valore assoluto da 1 E -38 a 1 E +38*
•*FLOAT (o DOUBLE PRECISION)*	*Numero reale con mantissa di precisione 15*	*valore 0 oppure valore assoluto da 1 E -38 a 1 E +38*
•*FLOAT(p)*	*Numero reale con mantissa di precisione p*	*p da 1 a 45*

Per i dati numerici la precisione p indica il numero massimo di cifre che il numero può contenere, esclusi il segno e il punto decimale. Per i numeri decimali il valore s indica il numero di cifre che seguono il punto decimale. I dati numerici floating point (numeri approssimati) sono memorizzati in forma esponenziale ; la precisione riguarda solo le cifre della mantissa.

La parola NUMERIC può essere usata al posto di DECIMAL.

La parola CHARACTER è equivalente a CHAR o CARACTER(1).

Il linguaggio SQL

Identificatori e tipi di dati

DECIMAL è equivalente a DECIMAL(15,0).
Altre abbreviazioni sono:
CHAR(n) invece di CHARACTER(n);
INT(p) invece di INTEGER(p);
DEC(p,s) invece di DECIMAL(p,s).
Occorre osservare che alcune versioni di SQL in specifici ambienti DBMS differiscono dallo standard nell'indicazione dei tipi di dati o nei termini utilizzati.
Nelle colonne della tabella gli attributi che hanno un valore non disponibile o non definito assumono il valore ***Null.*** *Il valore Null non è mai uguale a nessun altro valore: è diverso dal valore zero per i dati numerici e dalla stringa vuota('') per i dati alfanumerici. Negli ordinamenti il valore Null compare all'inizio delle sequenze crescenti e alla fine delle sequenze decrescenti.*
Nella scrittura delle condizioni possono essere usati gli operatori NOT,AND,OR.
Le costanti numeriche possono avere o non avere il segno + oppure -. Nei numeri decimali la parte intera è separata dalle cifre decimali attraverso il punto. I numeri in floating point (approssimati) sono rappresentati in forma esponenziale, per esempio -1.2 E+7.
Nelle operazioni di confronto i numeri sono controllati in accordo con il loro valore algebrico, mentre le stringhe sono confrontate carattere per carattere a partire da sinistra secondo il codice ASCII.
Tutti i risultati di confronti numerici o stringa che riguardano attributi con valore Null sono considerati sconosciuti , perché Null non è uguale , maggiore o minore di nessun numero.
Il linguaggio SQL può solo controllare la presenza o l'assenza di Null in una colonna.

Il linguaggio SQL

La definizione delle tabelle

Le tabelle vengono definite con il comando **CREATE TABLE** seguito dal nome della tabella e dall'elenco degli attributi. Per ogni attributo occorre specificare il nome e il tipo di dato.
Per esempio, per creare una tabella contenente i dati dei dipendenti di un'azienda (matricola, cognome e nome, codice fiscale, data di assunzione, filiale in cui lavora, funzione all'interno dell'azienda, livello, stipendio base, indirizzo con via, cap, città e provincia si deve usare il seguente comando:

```
CREATE TABLE Personale
(Matricola      char(5) not null,
Cognome          char(30),
Nome             char(20),
CodFisc          char(16) not null,
Assunto          date,
Filiale          smallint,
Funzione         char(15),
Livello          smallint;
StipBase         integer,
Via              char(25),
Cap              char(5),
Città            char(20),
Prov             char(2)    );
```

Il linguaggio SQL

La definizione delle tabelle

L'attributo Matricola è la chiave primaria della tabella.

Accanto alla definizione dell'attributo può essere specificata la clausola NOT NULL, per indicare che in tutte le righe della tabella quella colonna deve essere riempita con un valore non nullo, nelle operazioni di inserimento e di aggiornamento.

Si può utilizzare lo stesso nome di attributo in tabelle diverse; durante la gestione dei dati, poi, per distinguere gli attributi con nome uguale si usa la notazione:

Tabella.Attributo

La struttura della tabella può essere successivamente modificata con il comando **ALTER TABLE**, per aggiungere una nuova colonna **(ADD)** a quella già esistente , oppure per togliere una colonna **(DROP)** .

Per esempio, il comando seguente consente di inserire il nuovo attributo con la data di nascita del dipendete:

```
ALTER TABLE Personale
        ADD Nascita date;
```

Utilizzando l'opzione **Drop** si può togliere per esempio dalla tabella Personale l'attributo che si riferisce allo stipendio base:

```
ALTER TABLE Personale
        DROP StipBase;
```

L'istruzione CREATE INDEX

L'istruzione **CREATE INDEX** viene utilizzata per creare un nuovo indice su una tabella esistente, indicando il nome della tabella e il nome dell'attributo o degli attributi ai quali associare l'indice.

Se non si vuole che ci siano valori duplicati per l'attributo associato ad indice in righe diverse, occorre usare la clausola **UNIQUE**.

Per esempio il comando

CREATE UNIQUE INDEX Ipers
ON Personale (Cognome, Nome);

crea un indice di nome Ipers sulla tabella Personale secondo gli attributi Cognome e Nome.

Una tabella o un indice possono essere eliminati con il comando **DROP** , seguito dal nome della tabella o dell'indice.

Per esempio il comando seguente cancella la tabella Personale:

DROP TABLE Personale;

In modo analogo per cancellare l'indice di nome Ipers si usa il comando:

DROP INDEX Ipers ON Personale;

*I comandi illustrati finora rappresentano la parte del linguaggio SQL che fa riferimento alla categoria dei comandi definiti con la sigla **DDL** (Data Definition Language), cioè il linguaggio che consente di implementare il modello logico dei dati sulle memorie di massa del computer.*

Un file **indice** è una struttura dati realizzata per migliorare i tempi di ricerca dei dati. Se una tabella non ha indici, ogni ricerca obbliga il sistema a leggere tutti i dati presenti in essa. L'indice consente invece di ridurre l'insieme dei dati da leggere per completare la ricerca. Ad esempio, se si ha un insieme di dati disordinato, è possibile crearne un "indice" in ordine alfabetico, e sfruttare le proprietà dell'ordine alfabetico per arrivare prima al dato o ai dati cercati. Si potrebbe pensare, ad esempio, di applicare una **ricerca binaria** all'indice ordinato per reperire in tempi più brevi le informazioni richieste. Gli indici hanno anche degli effetti negativi in quanto rendono più lente le operazioni di inserimenti e modifica (update), ed aumentano l'uso della memoria di massa.

I comandi per la manipolazione dei dati

I valori degli attributi nelle righe della tabella, possono essere inseriti, aggiornati o cancellati rispettivamente con i comandi **INSERT,UPDATE** e **DELETE.**

Illustriamo l'uso di questi comandi con alcuni esempi.

Per inserire i valori di una nuova riga nella tabella Personale si usa il comando:

INSERT INTO PERSONALE
(Matricola, Cognome, Nome. CodFisc, Nascita, Assunto, Filiale, Funzione,Livello,StipBase Via, Cap, Citta, Prov)
Values('AB541','ROSSI','ROBERTO','RSSRRT66A26C034I','01/01/1966','01/09/1992','3','DOCENTE',7,2000 ,'VIA CIMAROSA, 99','03043', 'CASSINO','FR');

Per cambiare il livello al dipendente con matricola 'AA345' , occorre dare il comando

UPDATE Personale
SET Livello=6
WHERE Matricola='AA345';

È possibile cambiare anche più attributi con un solo comando Update elencandone i nomi e i nuovi valori dopo la parola Set, separati dalla virgola.

Per cancellare dalla tabella Personale i dati del dipendente con matricola ='AQ123', si usa il comando:

DELETE FROM Personale
WHERE Matricola ='AQ123';

È molto importante notare che l'uso della clausola Where (dove) nei comandi Update e Delete consente di **operare su gruppi di record**, cioè su righe , anziché su una sola riga per volta: basta indicare dopo Where una condizione che deve essere verificata dalle righe che si vogliono modificare o cancellare.

Per esempio, se si vuole aumentare del 5% lo stipendio di tutti i dipendenti sopra il quinto livello, occorre dare il comando:

UPDATE Personale
SET StipBase=StipBase*1.05
WHERE livello>5;

Il comando DELETE

Per eliminare dalla tabella Personale i dipendenti che hanno lo stipendio Base inferiore a 750, si usa il seguente comando:

DELETE FROM Personale
WHERE StipBase<750;

Nell'uso pratico del linguaggio le operazioni di inserimento, così come quelle di modifica e di cancellazione, vengono facilitate mediante l'uso di maschere video che guidano l'utente nelle operazioni di trattamento dei dati, all'interno di un ambiente software basato sull'uso di interfacce grafiche amichevoli. Quindi questi comandi di tipo **DML** (Data Manipulation Language) presenti all'interno del linguaggio SQL rimangono nascosti all'utente che può effettuare le operazioni di manipolazione senza conoscere la sintassi delle istruzioni.

Il Comando SELECT per porre interrogazioni

L'aspetto più importante del linguaggio SQL è costituito dalla possibilità di porre interrogazioni in modo molto semplice alla base di dati per ritrovare le informazioni che interessano.

Queste prestazioni sono fornite dal comando **SELECT,** che è nello stesso tempo molto potente e molto semplice da usare.

Con il comando Select vengono attivate le interrogazioni sulle relazioni e le operazioni relazionali per ottenere nuove tabelle. Inoltre l'utente viene liberato da tutti i problemi che riguardano le modalità e i percorsi per ritrovare i dati, cioè la loro collocazione fisica nelle memorie di massa: deve solo specificare al sistema quali dati vuole ottenere.

La struttura generale del comando Select è la seguente:

SELECT……
FROM……
WHERE…..

Accanto alla parola Select vengono indicati i nomi degli attributi (le colonne) da elencare (se è necessario elencare tutti gli attributi basta scrivere il segno asterisco * dopo la parola Select); dopo il From vengono indicati i nomi delle tabelle su cui deve operare il comando Select; dopo la clausola Where si specifica la condizione che deve essere soddisfatta dai campi delle righe:possono comparire anche più condizioni combinate con gli operatori AND,OR e NOT.

Il comando Select per porre interrogazioni

Per esempio nella tabella Personale vogliamo conoscere l'elenco formato dal cognome, nome e codice fiscale dei dipendenti con funzione di Impiegato, allora tale risultato si ottiene con il comando Select nella forma:

```
SELECT Cognome, Nome, CodFisc
    FROM Personale
   WHERE Funzione='Impiegato';
```

In questo primo esempio accanto alla parola Select sono stati specificati solo alcuni attributi tra quelli presenti nella tabella Personale.

Per richiedere tutti i dati dei dipendenti che abitano in provincia di Milano, si usa il comando Select nella forma:

```
SELECT  *
    FROM Personale
   WHERE Prov='Milano';
```

Il comando **Select** possiede due predicati **ALL** e **DISTINCT**.

Il predicato ALL indica la richiesta di ottenere come risultato dell'interrogazione tutte le righe che soddisfano alle condizioni contenute nel comando.

Questo predicato è di default, cioè se non viene fatta nessuna specificazione vengono visualizzate tutte le righe della tabella che rispondono alle condizioni poste.

Pertanto le due seguenti istruzioni Select sono equivalenti:

```
SELECT ALL  *                          SELECT  *
    FROM Personale        equivale a       FROM Personale
    WHERE Prov='MI';                       WHERE Prov='MI';
```

Se viene specificato il predicato DISTINCT le righe duplicate nella tabella risultante vengono ridotte a una.

Per esempio se si desidera ottenere l'elenco di tutte le professioni presenti tra i dipendenti della tabella Personale, specificando una sola volta il tipo di professione anche quando è riferita a più dipendenti, si deve usare la clausola DISTINCT prima dell'indicazione dell'attributo:

```
SELECT DISTINCT Funzione
  FROM Personale;
```

La clausola Distinct

Il comando SELECT senza la clausola DISTINCT

SELECT Funzione
FROM Personale;

produrrebbe l'elenco di tutte le professioni del dipendente ripetendo tante volte la stessa professione in righe diverse, tante quanti sono i dipendenti che svolgono quella funzione.

In presenza del predicato Distinct in un'interrogazione con Select che contiene più attributi , una riga può essere inclusa nella tabella risultante solo se la combinazione di valori provenienti da tutti gli attributi è univoca.

Prendiamo la nostra tabella Personale popolata con i dati sotto :

Personale

Matricola	Cognome	Nome	CodFisc	Nascita	Assunto	Filiale	Funzione	Livello	StipBase	Via	Cap	Citta	Prov
AAC51	VERDE	SERGIO	VRDSRG67B25M034F	25/02/1967	10/10/1998	2	ATA	5	1500	VIA VERDI, 33	20121	MILANO	MI
AB541	ROSSI	ROBERTO	RSSRRT66A26C034I	26/01/1966	01/09/1992	3	DOCENTE	7	2000	VIA CIMAROSA, 9	03043	CASSINO	FR
BB435	BIANCHI	MARCO	BNCMRC68C22B38HD	22/03/1968	19/11/1998	2	DOCENTE	7	1950	VIA MARCONI, 45	20121	MILANO	MI
CC72W	CASCARDI	VINCENZA	CSCVCN67A25V233F	25/01/1967	20/01/1994	3	ATA	5	1890	VIALE DANTE	03043	CASSINO	FR
CFG46	PASCALE	MARCO	PSCMRC58A22B043M	22/01/1958	01/12/1988	3	ATA	5	1990	VIALE MAZZINO, !	20121	MILANO	MI
PP454	PANNELLA	FRANCO	PNNFRN66A26MP34I	26/01/1967	02/10/1993	3	DIRIGENTE	8	2500	VIA MANZONI, 5	00118	ROMA	RM

Se creiamo la query per ottenere l'elenco di tutte le professioni presenti tra i dipendenti della tabella Personale ed usiamo la clausola DISTINCT, così come sotto riportato:

SELECT DISTINCT Funzione
FROM Personale;

La query sopra produrrà l'elenco di tutte le professioni dei dipendenti ripetuti una sola volta, ossia:

Funzione
ATA
DIRIGENTE
DOCENTE

Diversamente, il comando Select nella forma senza il Distinct produrrà il seguente elenco:

SELECT Funzione
FROM Personale;

Funzione
ATA
DOCENTE
DOCENTE
ATA
ATA
DIRIGENTE

Il comando Select con l'intestazione delle colonne

La tabella che si ottiene come risultato dell'interrogazione con Select normalmente possiede un'**intestazione delle colonne** che riporta i nomi degli attributi; se si vuole modificare tale intestazione , occorre dichiarare la stringa della nuova intestazione insieme alla clausola **AS.**

Per esempio per ottenere l'elenco delle diverse province da cui provengono i dipendenti della tabella Personale, intestando la colonna del risultato con il titolo Provincia , si deve scrivere il comando Select secondo la seguente sintassi:

SELECT DISTINCT *Prov* ***AS*** *Provincia*
FROM *Personale;*

Otteniamo il seguente risultato

Provincia
FR
MI
RM

Con il comando Select si può anche chiedere il calcolo di espressioni sugli attributi presenti nella tabella; la tabella risultante contiene una colonna aggiuntiva con i risultati del calcolo per ogni riga. Questa nuova colonna viene intestata con una descrizione opportuna utilizzando la parola As.

Per esempio il comando seguente visualizza, accanto agli stipendi attuali, quali sarebbero i nuovi stipendi base dei dipendenti aumentati del 5%, senza modifica dell'importo di ciascuno stipendio nelle righe della tabella:

```
SELECT Cognome, Nome, StipBase As Attuale,
    StipBase*1.05 AS Nuovo
FROM Personale;
```

Riportiamo la tabella prodotta dal comando sopra:

Cognome	Nome	Attuale	Nuovo
VERDE	SERGIO	1500	2250
ROSSI	ROBERTO	2000	3000
BIANCHI	MARCO	1950	2925
CASCARDI	VINCENZA	1890	2835
PASCALE	MARCO	1990	2985
PANNELLA	FRANCO	2500	3750

Il comando Select con l'utilizzo di parametri

In alcuni degli esempi precedenti e in quelli presentati nei paragrafi successivi, per rendere più semplice l'illustrazione dei comandi di SQL, vengono utilizzati valori costanti nelle condizioni scritte dopo Where, come nel caso dell'interrogazione di inizio paragrafo:

```
SELECT Cognome, Nome , CodFisc
   FROM Personale
   WHERE Funzione="DOCENTE";
```

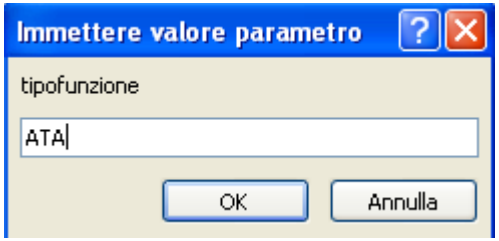
Risultato →

Cognome	Nome	CodFisc
ROSSI	ROBERTO	RSSRRT66A26C
BIANCHI	MARCO	BNCMRC68C22

Il risultato del comando sopra, come risulta dalla figura , rende l'idea delle condizioni scritte con valori costanti. In realtà a volte è utile utilizzare un parametro al posto della costante (DOCENTE), in tal modo sarà possibile utilizzare la stessa interrogazione più volte per funzioni diverse dei dipendenti. Il nome del parametro può essere inserito all'interno degli apici singoli o tra parentesi quadre , dipende dal software utilizzato. Quindi se vogliamo parametrizzare l'interrogazione possiamo scrivere le istruzioni riportate di seguito:

```
SELECT Cognome, Nome , CodFisc
   FROM Personale
   WHERE Funzione=[tipofunzione];
```

INPUT →



RISULTATO →

Cognome	Nome	CodFisc
VERDE	SERGIO	VRDSRG67B25I
CASCARDI	VINCENZA	CSCVCN67A25
PASCALE	MARCO	PSCMRC58A22

Le operazioni relazionali nel linguaggio SQL

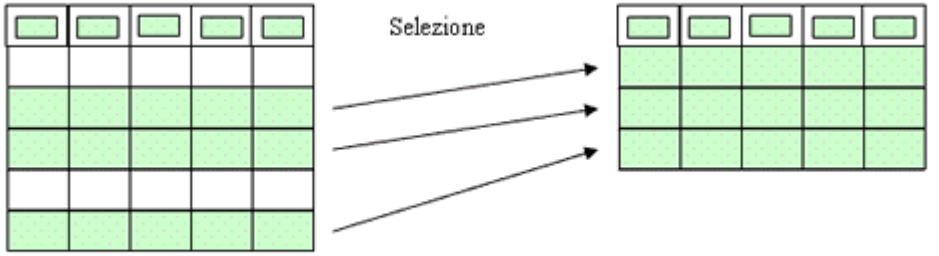
Le operazioni di selezione, proiezione e congiunzione su una base di dati relazionale vengono realizzate in pratica attraverso il comando Select, secondo le diverse forme consentite dalla sintassi di questo comando. Se si vuole rispettare una delle regole del modello relazionale che non consente la presenza di righe uguali all'interno della stessa tabella, basta porre accanto alla parola Select la clausola Distinct: con questa specificazione la tabella ottenuta non contiene righe duplicate.

La selezione e la proiezione sono già state utilizzate in pratica negli esempi d'uso del comando Select nelle pagine precedenti.

L'operazione di selezione (select) nel linguaggio SQL

L'operazione di **selezione** , che consente di ricavare da una relazione un'altra relazione contenente solo le righe che soddisfano ad una certa condizione, viene realizzata nel linguaggio SQL utilizzando la clausola **Where** del comando Select.

Per esempio per ottenere l'elenco con tutti i dati dei dipendenti che svolgono la funzione di Dirigente, si opera una selezione sulla tabella Personale estraendo le righe per le quali l'attributo Funzione contiene il valore 'Dirigente' . Quindi l'interrogazione viene codificata in SQL con il comando:



SELECT *

FROM Personale

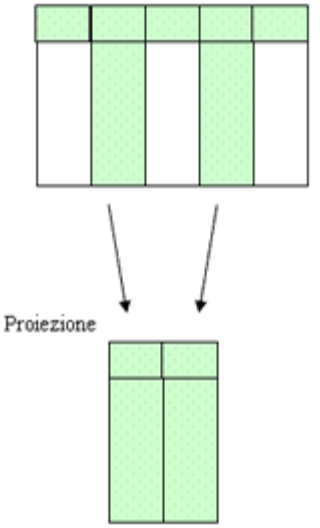
WHERE Funzione= "Dirigente";

Il cui risultato è rappresentato nella tabella sotto

Matricola	Cognom	Nome	CodFisc	Nascita	Assunto	Filiale	Funzione	Livello	StipBase	Via	Cap	Citta	Prov
PP454	PANNELLA	FRANCO	PNNFRN66A26MP	26/01/1967	02/10/1993	3	DIRIGENTE	8	2500	VIA MANZONI,	00118	ROMA	RM

Le operazioni di proiezione (project) del linguaggio SQL

L' operazioni di **proiezione**, che permette di ottenere una relazione contenente solo alcuni attributi della relazione di partenza, si realizza indicando accanto alla parola Select l'elenco degli attributi richiesti. Per esempio per ottenere l'elenco di tutti i dipendenti con cognome, nome e codice fiscale, si deve effettuare una proiezione sulla tabella Personale estraendo soltanto le colonne corrispondenti agli attributi richiesti.



Questa operazione viene codificata in linguaggio SQL con il comando:

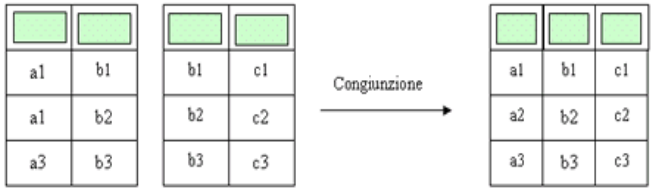
```
SELECT Cognome, Nome, CodFisc
  FROM Personale;
```

Cognome	Nome	CodFisc
VERDE	SERGIO	VRDSRG67B25M0
ROSSI	ROBERTO	RSSRRT66A26C03
BIANCHI	MARCO	BNCMRC68C22B3
CASCARDI	VINCENZA	CSCVCN67A25V23
PASCALE	MARCO	PSCMRC58A22B04
PANNELLA	FRANCO	PNNFRN66A26MP

Le operazioni di congiunzione (o join) del linguaggio SQL

Il comando Select può operare su più tabelle , indicandone i nomi (separati da virgola) dopo la parola FROM ; scrivendo poi dopo la parola Where i nomi degli attributi che si corrispondono nelle due tabelle (legati tra loro con il segno =), si realizza in pratica l'operazione di **congiunzione** *di due tabelle secondo un attributo comune. Per esempio si supponga di aver creato accanto alla tabella Personale anche la tabella Dipendente con il comando*

```
CREATE TABLE Dipendente
       (CodFil Smallint,
        Descrizione char(20),
        Indirizzo  char(25));
```

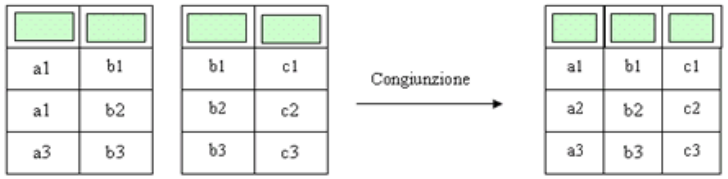


Contenente per ogni dipendenza il codice, usato nei dati dipendenti, la descrizione e l'indirizzo della filiale. L'attributo CodFil è la chiave primaria della tabella. Vediamo la tabella popolata con alcuni campi (figura a destra).

Personale | Dipendente

CodFil	Descrizione	Indirizzo
1	Sede di Milanc	Viale America,
2	Sede di Frosin	P.zza Garibaldi
3	Sede di Roma	Via Mazzini, 34
4	Sede di Cassin	Via Verdi, 32

Le operazioni di congiunzione (o join) del linguaggio SQL

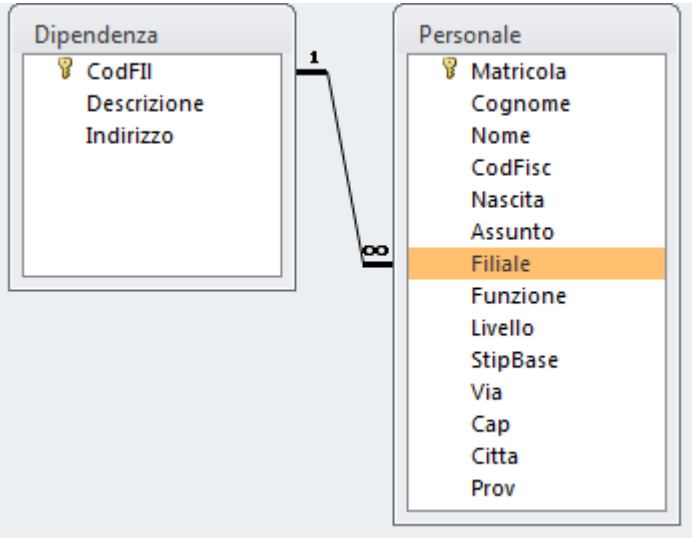


Tra la relazione *Dipendenza* e la relazione *Personale* viene stabilità un'*associazione* **uno** a **molti** e *l'attributo Filiale* nella tabella *Persona* diventa ***chiave esterna***. Per ottenere l'elenco di tutti i dipendenti con la descrizione e l'indirizzo della filiale dove lavorano, occorre effettuare la congiunzione delle tabelle **Personale** e **Dipendenza** secondo gli attributi comuni **Filiale** e **CodFil**.

In linguaggio SQL *:* ***SELECT ****

FROM Personale, Dipendenza

WHERE Filiale=CodFil;



Per maggiore chiarezza si può anche scrivere nel modo seguente, esplicitando il nome della tabella insieme ai nomi degli attributi comuni:

SELECT *

FROM Personale, Dipendenza

WHERE Personale.Filiale = Dipendenza.CodFil;

Otteniamo la seguente tabella che contiene l'elenco di tutti i dipendenti con la descrizione e l'indirizzo della filiale dove lavorano.

Matricol	Cognom	Nome	CodFisc	Nascita	Assunto	Filial	Funzioni	Live	StipB	Via	Cap	Citta	Prc	CodFIl	Descrizione	Indirizzo
AAC51	VERDE	SERGIO	VRDSRG67B25M03	25/02/1967	10/10/1998	2	ATA	5	1500	VIA VERDI, 33	20121	MILANO	MI	2	Sede di Frosin	P.zza Garibaldi
BB435	BIANCHI	MARCO	BNCMRC68C22B38	22/03/1968	19/11/1998	2	DOCENTE	7	1950	VIA MARCONI,	20121	MILANO	MI	2	Sede di Frosin	P.zza Garibaldi
AB541	ROSSI	ROBERTO	RSSRRT66A26C034	26/01/1966	01/09/1992	3	DOCENTE	7	2000	VIA CIMAROSA	03043	CASSINC	FR	3	Sede di Roma	Via Mazzini, 34
CC72W	CASCARDI	VINCENZA	CSCVCN67A25V23	25/01/1967	20/01/1994	3	ATA	5	1890	VIALE DANTE	03043	CASSINC	FR	3	Sede di Roma	Via Mazzini, 34
CFG46	PASCALE	MARCO	PSCMRC58A22B04	22/01/1958	01/12/1988	3	ATA	5	1990	VIALE MAZZIN	20121	MILANO	MI	3	Sede di Roma	Via Mazzini, 34
PP454	PANNELLA	FRANCO	PNNFRN66A26MP	26/01/1967	02/10/1993	3	DIRIGENTE	8	2500	VIA MANZONI,	00118	ROMA	RM	3	Sede di Roma	Via Mazzini, 34

Le operazioni di congiunzione (o join) del linguaggio SQL

Poiché nella condizione scritta accanto a Where viene usato il segno uguale, la congiunzione precedente fornisce un esempio di **equi-join**. Cioè vengono combinate solo le righe per le quali si possono trovare valori uguali negli attributi che si corrispondono. Nella versione del linguaggio SQL per Access in ambiente Windows l'equi-join viene indicato con **inner join** (join interno); inoltre si possono realizzare anche **join esterni** (outer join) con *left join* e *right join*.

Self-join

Abbiamo già detto che il **self-join** è una congiunzione con la quale vengono combinate righe di una tabella con righe della stessa tabella, questa operazione ha senso se sono presenti valori corrispondenti per attributi , cioè due attributi con lo stesso dominio. Ad esempio, supponiamo di aggiungere agli attributi della tabella Personale un nuovo attributo denominato Dirigente adatto a contenere, per ogni dipendente, il codice della persona che svolge il ruolo di dirigente rispetto a quel dipendente. Le informazioni sui dirigenti sono contenute nella stessa tabella dei dipendenti, poiché sono anch'essi dipendenti dell'azienda: pertanto se si vogliono conoscere le informazioni dei dipendenti con il dirigente a cui fanno capo, occorre fare una congiunzione della tabella Personale con se stessa, cioè un **self-join**.

Relazioni | Query1 | **Personale**

Matricola	Cognome	Nome	CodFisc	Nascita	Assunto	Filiale	Funzione	Livell	StipBase	Via	Cap	Citta	Prov	Dirigen
AAC51	VERDE	SERGIO	VRDSRG67B25M034F	25/02/1967	10/10/1998	2	ATA	5	1500	VIA VERDI, 33	20121	MILANO	MI	PANNELLA
AB541	ROSSI	ROBERTO	RSSRRT66A26C034I	26/01/1966	01/09/1992	3	DOCENTE	7	2000	VIA CIMAROSA, 9	03043	CASSINO	FR	PANNELLA
BB435	BIANCHI	MARCO	BNCMRC68C22B38HD	22/03/1968	19/11/1998	2	DOCENTE	7	1950	VIA MARCONI, 45	20121	MILANO	MI	PANNELLA
CC72W	CASCARDI	VINCENZA	CSCVCN67A25V233F	25/01/1967	20/01/1994	3	ATA	5	1890	VIALE DANTE	03043	CASSINO	FR	PANNELLA
CFG46	PASCALE	MARCO	PSCMRC58A22B043M	22/01/1958	01/12/1988	3	ATA	5	1990	VIALE MAZZINO, !	20121	MILANO	MI	PANNELLA
PP454	PANNELLA	FRANCO	PNNFRN66A26MP34I	26/01/1967	02/10/1993	3	DIRIGENTE	8	2500	VIA MANZONI, 5	00118	ROMA	RM	PANNELLA

Per risolvere questo problema si devono usare gli alias per il nome della tabella, cioè indicare con due nomi diversi la stessa tabella, utilizzando la parola **AS** nella clausola **FROM** del comando SELECT, come mostrato nell'esempio. Con questa interrogazione si ottiene per ciascun dipendente il cognome e nome insieme al cognome del dirigente:

```
SELECT  Pers.Cognome, Pers.Nome, Dirig.Cognome
    FROM Personale AS Pers, Personale AS Dirig
    WHERE Pers.Dirigente=Dirig.Cognome;
```

Il risultato ottenuto è la tabella a destra contenente il Cognome del dirigente di tutti i dipendenti, ed è intuitivo il fatto che esiste un solo dirigente.

Pers.Cognom	Nome	Dirig.Cognome
PANNELLA	FRANCO	PANNELLA
PASCALE	MARCO	PANNELLA
CASCARDI	VINCENZA	PANNELLA
BIANCHI	MARCO	PANNELLA
ROSSI	ROBERTO	PANNELLA
VERDE	SERGIO	PANNELLA

Le operazioni del linguaggio SQL

Dopo aver visto come si rappresentano nel linguaggio SQL le operazioni fondamentali del modello relazionale, si può facilmente comprendere a questo punto come sia realizzabile la combinazione di diverse operazioni (proiezione e selezione, congiunzione e selezione, congiunzione e proiezione, congiunzione, selezione e proiezione) usando la struttura Select……From…..Where…..secondo tutte le possibili varianti.

Per esempio se si vuole ottenere l'elenco dei dipendenti che hanno la funzione di Docente, con cognome, nome, descrizione e indirizzo della Filiale dove lavorano, occorre dapprima operare una selezione su Personale per Funzione uguale al valore 'Docente'; poi si deve effettuare una congiunzione della tabella ottenuta su Filiale e di Dipendenza su CodFil ed infine sulla nuova tabella ottenuta si applica una proiezione sugli attributi Cognome, Nome, Descrizione, Indirizzo.

In presudocodifica l'interrogazione si rappresenta in questo modo:

Proiezione di

(Congiunzione di

(Selezione di Personale per Funzione='Docente')

su Filiale e di Dipendenza su CodFil)

Su Cognome, Nome, Decrizione, Indirizzo

Il comando viene codificato nel linguaggio SQL con il seguente formato per Select:

```
SELECT Cognome, Nome, Descrizione, Indirizzo
    FROM  Personale, Dipendenza
    WHERE Filiale=CodFIl
    AND Funzione='Docente';
```

La prima condizione scritta dopo Where (Filiale=CodFil) serve a specificare l'uguaglianza tra gli attributi comuni delle tabelle coinvolte nella congiunzione, la seconda condizione (Funzione='Docente') specifica il criterio per operare la selezione sulle righe della tabella risultante dalla congiunzione. Le due condizioni sono combinate tramite l'operatore AND. La codifica in SQL diventa più precisa facendo precedere il nome delle tabelle ai nomi degli attributi, come mostrato nella versione seguente:

```
SELECT Personale.Cognome, Personale.Nome, Dipendenza.Descrizione, Dipendenza.Indirizzo
    FROM  Personale, Dipendenza
    WHERE Personale.Filiale=Dipendenza.CodFIl
    AND Personale.Funzione='Docente';
```

Cognom	Nome	Descrizione	Indirizzo
BIANCHI	MARCO	Sede di Frosin	P.zza Garibaldi
ROSSI	ROBERTO	Sede di Roma	Via Mazzini, 34

La clausola INTO

Abbiamo detto che le operazioni di selezione, proiezione, congiunzione e le combinazioni tra esse servono ad ottenere una nuova relazione dalla relazione di partenza. Se si vuole conservare la tabella risultante dall'operazione, occorre aggiungere al comando Select la clausola **INTO** seguita dal nome da assegnare alla nuova tabella.

Per esempio se si desidera creare una nuova tabella di nome *Manager* contenente le informazioni per i soli dipendenti che hanno la funzione di Dirigente, occorre usare un comando come il seguente:

```
SELECT * INTO Manager
    FROM  Personale
    WHERE Funzione='Docente';
```

Query3 | Manager | Personale

Matricola	Cognome	Nome	CodFisc	Nascita	Assunto	Filiale	Funzione	Livello	StipBase	Via	Cap	Citta	Prov
AB541	ROSSI	ROBERTO	RSSRRT66A26C034I	26/01/1966	01/09/1992	3	DOCENTE	7	2000	VIA CIMAROSA, 99	03043	CASSINO	FR
BB435	BIANCHI	MARCO	BNCMRC68C22B38	22/03/1968	19/11/1998	2	DOCENTE	7	1950	VIA MARCONI, 45	20121	MILANO	MI

Le colonne della nuova tabella presentano gli stessi nome, formati e dimensioni degli attributi della tabella originale.

Se invece si vogliono aggiungere le righe della tabella risultante alle righe di una tabella già esistente, si deve usare il comando **Insert Into** seguito dal nome della tabella e dalla frase Select che determina le righe da aggiungere nella tabella.

Se si vogliono aggiungere alla tabella Personale le righe provenienti da un'altra tabella contenente i dati dei nuovi assunti, il comando assume il formato:

```
INSERT INTO PERSONALE
  SELECT *
    FROM NuoviAssunti;
```

La funzione di aggregazione

All'interno del comando *Select* possono essere usate funzioni predefinite che agiscono sui valori contenuti in insiemi di righe della tabella e che per questo motivo si chiamano **funzioni di aggregazione.**

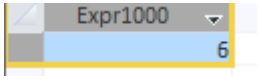
- **Funzione COUNT**

La funzione **COUNT** conta il numero di righe presenti in una tabella. La sintassi del linguaggio SQL richiede di specificare come argomento della funzione il nome di un attributo oppure il carattere * (asterisco): nel primo caso non vengono conteggiate le righe che hanno valore NULL nella colonna dell'attributo specificato; nel secondo caso, indicando l'asterisco, la funzione COUNT(*) calcola il numero delle righe della tabella. Incluse quelle con campi di tipo NULL.

In sostanza la funzione COUNT(*) serve per ottenere la cardinalità (numero di righe o tuple) di una relazione. La funzione calcola solo il numero delle righe, indipendentemente dai valori in esse memorizzati.

Il seguente comando restituisce il numero di tutte le righe presenti nella tabella Personale:

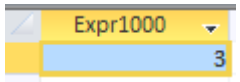
```
SELECT COUNT (*)
  FROM Personale;
```



Se utilizzata in un comando Select che contiene il controllo dei una condizione scritta dopo Where, la funzione COUNT restituisce il numero di righe che soddisfano alla condizione specificata:

La seguente interrogazione restituisce il numero dei dipendenti della provincia di Milano presenti nella tabella Personale:

```
SELECT COUNT (*)
   FROM Personale
  WHERE Prov='MI';
```



La funzione di aggregazione COUNT(DISTINCTx)

La stessa funzione nella versione COUNT(DISTINCT x) consente di ottenere il numero dei valori diversi tra loro nella colonna x che soddisfano alla condizione scritta dopo Where. Per esempio se si vuole conoscere a quanti livelli appartengono i dipendenti che abitano in provincia di Milano, occorre dare il comando:

```
SELECT COUNT(DISTINCT livello)
  FROM Personale
  WHERE Prov='MI';
```

La clausola DISTINCT non può essere usata nel formato con l'asterisco COUNT(*).

Il risultato del conteggio può essere anche scritto con un'opportuna intestazione, come già visto in precedenza, aggiungendo la clausola As seguita dalla stringa da assegnare.

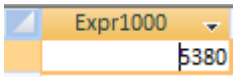
```
SELECT COUNT(DISTINCT livello) AS Livelli
  FROM Personale
  WHERE Prov='MI';
```

Funzione SUM

La funzione **SUM** restituisce la somma di tutti i valori contenuti in una colonna specificata come argomento della funzione; naturalmente l'attributo utilizzato nel calcolo deve essere di tipo numerico.

La funzione SUM considera i record contenenti campi di tipo Null come aventi valore 0. Se nel comando Select è presente il controllo di una condizione con Where, la funzione calcola la somma dei valori contenuti in una colonna solo per le righe che soddisfano alla condizione specificata dopo la clausola Where. Per esempio se si vuole ottenere la somma degli stipendi base dei dipendenti che appartengono al livello 5, occorre usare il comando Select nella forma:

```
SELECT SUM (StipBase)
   FROM Personale
   WHERE Livello=5;
```



Anche con questa funzione si può usare, se necessario, la parola Distinct prima del nome dell'attributo, se si vogliono eliminare dal calcolo della somma i valori duplicati presenti nella colonna specificata.

L'argomento della funzione SUM può essere anche un'espressione numerica contenente i nomi di attributi di tipo numerico. Si supponga per esempio di avere una tabella delle fatture che contiene tra gli attributi, per ciascuna fattura, il prezzo unitario dei prodotti e la quantità ordinata. Il totale delle fatture viene calcolato con il seguente comando Select:

```
SELECT SUM (PrezUnit *Qta) AS Totale
       FROM Fatture;
```

La funzione AVG

In modo analogo agisce la funzione AVG (Average=media), per calcolare la media dei valori numerici contenuti in una determinata colonna di una tabella, con l'eventuale aggiunta dell'opzione Distinct. La funzione AVG non include nel calcolo i valori di tipo Null presenti nella colonna. Nell'esempio seguente viene calcolato lo stipendio medio dei dipendenti che svolgono la funzione di docenti:

```
SELECT AVG (StipBase)
        FROM Personale
        WHERE Funzione='Docente';
```

Expr1000
1975

Le funzioni MIN e MAX

Le funzioni MIN e **MAX** restituiscono rispettivamente il valore minimo e il valore massimo tra i valori della colonna di una tabella specificata come argomento della funzione; nel caso sia specificata la clausola WHERE calcolano il valore minimo e massimo dei valori di una colonna considerando solo le righe che soddisfano la condizione.

Le funzioni *MIN e MAX* consentono di determinare i valori minimi e massimi anche per campi di tipo **carattere**.

Per esempio utilizzando un attributo numerico come lo stipendio base si calcolano i valori minimo e massimo degli stipendi di tutti i dipendenti.

```
SELECT MIN(StipBase) AS StipendioMin, MAX(StipBase) AS StipendioMax
    FROM Personale;
```

StipendioMin	StipendioMax
1500	2500

Il seguente esempio di interrogazione invece restituisce il primo cognome e l'ultimo nell'elenco dei dipendenti, specificando come argomento delle funzioni l'attributo Cognome che è di tipo carattere:

```
SELECT MIN(Cognome), MAX(Cognome)
    FROM Personale;
```

Expr1000	Expr1001
BIANCHI	VERDE

Se invece vogliamo rappresentare la stringa NomeMin e NomeMax nell'intestazione della tabella, allora l'interrogazione diventa:

```
SELECT MIN(Cognome) AS NomeMin, MAX(Cognome) AS NomeMax
    FROM Personale;
```

NomeMin	NomeMax
BIANCHI	VERDE

Anche le funzioni Min e Max ignorano i campi con valore Null e possono avere come argomento un'espressione anziché il nome di un attributo.

Ordinamenti e raggruppamenti

Nel comando Select si può usare la clausola **ORDER BY** per ottenere i risultati di un'interrogazione ordinati secondo i valori contenuti in una o più colonne, tra quelle elencate accanto alla parola Select. Utilizzando la parola chiave **ASC** e **DESC,** si possono ordinare in modo crescente e decrescente i dati di un attributo. Di default l'ordinamento è di tipo crescente, quindi va specificata la parola DESC solo se si desidera l'ordinamento decrescente.

Il seguente esempio riproduce in output l'elenco alfabetico dei dipendenti, con cognome, nome, codice fiscale e data di nascita:

Cognome	Nome	CodFisc	Nascita
BIANCHI	MARCO	BNCMRC68C22B38HD	22/03/1968
CASCARDI	VINCENZA	CSCVCN67A25V233F	25/01/1967
PANNELLA	FRANCO	PNNFRN66A26MP34I	26/01/1967
PASCALE	MARCO	PSCMRC58A22B043M	22/01/1958
ROSSI	ROBERTO	RSSRRT66A26C034I	26/01/1966
VERDE	SERGIO	VRDSRG67B25M034F	25/02/1967

```
SELECT Cognome, Nome, CodFisc, Nascita
    FROM  Personale
    ORDER BY Cognome, Nome;
```

L'ordinamento viene fatto su due attributi: A parità di cognome i dipendenti vengono ordinati per nome. Se i tipi di ordinamento richiesti riguardano più attributi e sono diversi, occorre aggiungere la parola DESC accanto agli attributi per i quali si vuole l'ordinamento decrescente.

L'esempio seguente serve a produrre l'elenco dei dipendenti in ordine decrescente di stipendio base e, a parità di stipendio, in ordine di cognome:

Cognome	StipBase
PANNELLA	2500
ROSSI	2000
PASCALE	1990
BIANCHI	2000
CASCARDI	1890
VERDE	1500

```
SELECT Cognome, StipBase
    FROM  Personale
    ORDER BY StipBase DESC, Cognome;
```

La clausola Order By è in genere l'ultimo elemento di un comando SQL.

Negli ordinamenti il valore Null compare all'inizio delle sequenze crescenti e alla fine delle sequenze decrescenti.

Con l'uso delle funzioni di aggregazione è possibile estendere la struttura del comando Select con l'aggiunta della clausola **GROUP BY** per raggruppare un insieme di righe aventi lo stesso valore nelle colonne indicate: ***questa opzione produce una riga di risultati per ogni raggruppamento.*** Se nel comando viene inserita una funzione di aggregazione, come Sum o Count, per ciascuna riga della tabella risultante viene prodotto un valore di raggruppamento. Il comando seguente serve ad ottenere la lista delle funzioni dei dipendenti con la somma degli stipendi e il numero dei dipendenti appartenenti alle diverse funzioni:

```
SELECT Funzione, SUM (StipBase) AS StipendioBase, COUNT (*) AS NumeroDip
    FROM Personale
    GROUP BY   Funzione;
```

Funzione	StipendioBas	NumeroDip
ATA	5380	3
DIRIGENTE	2500	1
DOCENTE	4000	2

Raggruppamenti

Se si utilizza una clausola **GROUP BY** , tutti gli attributi che compaiono nella lista accanto alla parola SELECT devono essere inclusi nella clausola GROUP BY oppure devono essere argomenti di una funzione di aggregazione.

Il seguente esempio produce l'elenco di livelli esistenti tra i dipendenti che svolgono la funzione di Docente con il numero di dipendenti per ciascun livello:

```
SELECT Livello, COUNT (Livello) AS Conteggio
    FROM PERSONALE
    WHERE Funzione='Docente'
    GROUP BY Livello;
```

Livello	Conteggio
7	2

La struttura del comando Select con raggruppamenti può essere ulteriormente ampliata con la clausola **HAVING** con la quale è possibile sottoporre al controllo di una o più condizioni i gruppi creati con la clausola Group By.

La condizione scritta dopo **Having** normalmente controlla il valore restituito dalle funzioni di aggregazione (Count, Sum, Avg, Min, Max).

Il seguente esempio presenta l'uso del comando Select per ottenere la lista delle funzioni dei dipendenti con lo stipendio medio per ciascuna funzione, dopo aver raggruppato i dipendenti per funzione, purché i dipendenti con quella funzione siano più di 1:

```
SELECT Funzione, AVG(StipBase) AS MediaStipBase
    FROM Personale
    GROUP BY Funzione
    HAVING COUNT (*) >1;
```

Funzione	MediaStipBase
ATA	1793,33333333333
DOCENTE	2000

In genere quindi la clausola Having viene usata insieme a Group By: dopo che Group By ha formato i raggruppamenti di righe, Having serve a visualizzare le righe di raggruppamento che soddisfano alle condizioni scritte accanto a Having.

Se l'istruzione Select contiene la clausola Where, i valori vengono raggruppati dopo aver operato la selezione sulle righe che rispettano le condizioni scritte accanto a Where.

Con il seguente comando si ottiene l'elenco delle filiali nelle quali ci sono più di 1 dipendente con la funzione ATA:

```
SELECT Filiale, COUNT (Filiale) AS Conteggio
   FROM Personale
   WHERE Funzione="ATA"
   GROUP BY Filiale
   HAVING COUNT (*)>1;
```

Filiale	Conteggio
3	2

Le condizioni di ricerca

Le **condizioni di ricerca** sono utilizzate insieme alla clausola **Where** e **Having** per determinare i criteri di selezione rispettivamente delle righe dei raggruppamenti. Nella scrittura delle condizioni si usano i segni di confronto ***=,<,>,<>,>=,<=.***

Una condizione di ricerca è costruita anche mettendo insieme più condizioni legate tra loro con gli operatori **AND** e **OR** , precedute eventualmente dall'operatore **NOT**. L'ordine di applicazione degli operatori è il seguente: NOT viene applicato prima di AND e AND prima di OR.

Le condizioni di ricerca possono utilizzare anche altre parole del linguaggio SQL che indicano operatori o predicati, con i quali è possibile rendere ancora più raffinate le interrogazioni alla base di dati.

- **BETWEEN**

L'operatore **BETWEEN** (Tra) controlla se un valore è compreso all'interno di un intervallo di valori, inclusi gli estremi. È possibile specificare, anteponendolo a Between , anche l'operatore logico **NOT** per valutare la condizione opposta, cioè per controllare se il valore non rientra nell'intervallo specificato.

Per ottenere l'elenco dei dipendenti (con cognome, nome e funzione) che hanno lo stipendio base tra 1800 e 2000 euro, si può usare la parola Between nella scrittura della condizione dopo **Where :**

```
SELECT Cognome, Nome, Funzione,StipBase
   FROM Personale
    WHERE StipBase BETWEEN 1800  AND 2000;
```

Cognome	Nome	Funzione	StipBase
ROSSI	ROBERTO	DOCENTE	2000
BIANCHI	MARCO	DOCENTE	2000
CASCARDI	VINCENZA	ATA	1890
PASCALE	MARCO	ATA	1990

- **IN**

L'operatore **IN** controlla se un valore appartiene ad un insieme specificato di valori, cioè è possibile richiedere le righe che hanno i valori di un attributo compresi in una lista di valori indicati dopo la parola *In a*ll'interno della condizione scritta dopo *Where* .

Per ottenere tutti i dati dei dipendenti che risiedono nelle province di Milano e Frosinone , si usa il comando Select nella forma:

```
SELECT *
   FROM Personale
    WHERE Prov IN ('MI','FR');
```

Matricola	Cognome	Nome	CodFisc	Nascita	Assunto	Filiale	Funzione	Livello	StipBase	Via	Cap	Citta	Prov	Dirig
AAC51	VERDE	SERGIO	VRDSRG67B25M034F	25/02/1967	10/10/1998	2	ATA	5	1500	VIA VERDI, 33	20121	MILANO	MI	PANNE
AB541	ROSSI	ROBERTO	RSSRRT66A26C034I	26/01/1966	01/09/1992	3	DOCENTE	7	2000	VIA CIMAROSA, 9	03043	CASSINO	FR	PANNE
BB435	BIANCHI	MARCO	BNCMRC68C22B38HD	22/03/1968	19/11/1998	2	DOCENTE	7	2000	VIA MARCONI, 45	20121	MILANO	MI	PANNE
CC72W	CASCARDI	VINCENZA	CSCVCN67A25V233F	25/01/1967	20/01/1994	3	ATA	5	1890	VIALE DANTE	03043	CASSINO	FR	PANNE
CFG46	PASCALE	MARCO	PSCMRC58A22B043M	22/01/1958	21/12/1988	3	ATA	5	1990	VIALE MAZZINO,	20121	MILANO	MI	PANNE

Le condizioni di ricerca

- **LIKE**

L'operatore **LIKE** confronta il valore di un attributo di tipo carattere con un modello di stringa che può contenere jolly (o metacaratteri).
I caratteri jolly sono:
_(underscore) per indicare un singolo carattere qualsiasi in quella posizione della stringa;
%(percento) per indicare una sequenza qualsiasi di caratteri in quella posizione della stringa.
***(asterisco)** per indicare una sequenza qualsiasi di caratteri in quella posizione della stringa.

Per esempio:
LIKE 'xyz%' vengono ricercate tutte le stringhe che iniziano con i caratteri 'xyz';
LIKE '%xyz' serve a ricercare tutte le stringhe che finiscono con i caratteri 'xyz';
LIKE '%xyz%' per tutte le stringhe che contengono al loro interno i caratteri 'xyz';
LIKE '_xyz' controlla le stringhe di 4 caratteri che finiscono con 'xyz';
LIKE 'xyz*' vengono ricercate tutte le stringhe che iniziano con i caratteri 'xyz'; questo carattere jolly è utilizzato in ambiente Access.

L'operatore **Like** utilizzato con un modello di stringa che non contiene caratteri jolly è del tutto equivalente all'operatore =.
Con il comando Select scritto nella forma seguente, è possibile ottenere il cognome e la filiale dei dipendenti con il cognome che inizia con 'Ros' (Rossi, Rossini, Rosi, ...);

```
SELECT Cognome, Filiale
   FROM  Personale
    WHERE Cognome LIKE  'Ros*';
```

Cognome	Filiale
ROSSI	3
ROSSINI	3

Anche in questo caso, così come per l'operatore IN, si può usare l'operatore NOT prima di LIKE per indicare criteri di ricerca opposti.

Le condizioni di ricerca

- **IS NULL**

Il predicato **IS NULL**, confronta il valore in una colonna con il valore *Null*. L'uso di questo predicato è il solo modo per controllare la presenza del valore *Null* in una colonna. È Possibile inserire l'operatore di negazione NOT per valutare la condizione opposta , in altre parole per controllare se un attributo non ha valore *Null.*

L'operatore *IS* viene utilizzato solo con la parola *Null.*

Per esempio se si vuole ottenere l'elenco con cognome e nome dei dipendenti per i quali è indicata la provincia nella tabella *Personale, si deve scrivere la seguente istruzione Select* **.**

```
SELECT Cognome, Nome
    FROM Personale
    WHERE  Prov IS NOT NULL;
```

Cognome	Nome
VERDE	SERGIO
ROSSI	ROBERTO
BIANCHI	MARCO
CASCARDI	VINCENZA
PASCALE	MARCO
PANNELLA	FRANCO
ROSSINI	MARCO

I comandi per la sicurezza

L'amministratore della base di dati, o comunque chi crea le relazioni del database, può stabilire anche il diritto di accesso per utenti specifici o per tutti gli utenti, nel caso di accessi multipli alle tabelle del database.

Il comando **GRANT** concede i permessi, specificando il tipo di accesso, le tabelle sulle quali è consentito l'accesso e l'elenco degli utenti ai quali è permesso di accedere.

Il tipo di accesso può riguardare per esempio il diritto di modifica della struttura delle tabelle con l'aggiunta di nuove colonne, oppure di modifica dei dati contenuti nella tabella, oppure l'uso del comando *Select.*

Per concedere il diritto di modifica sulla tabella dei dipendenti agli utenti denominati con User1 e User2, si deve usare il comando Grant nella forma:

```
GRANT UPDATE
    ON Personale
    TO User1, User2;
```

I comandi per la sicurezza

La revoca dei permessi con annullamento dei diritti di accesso viene effettuato con il comando **REVOKE** che ha una sintassi analoga a quella del comando Grant **:**

REVOKE UPDATE
ON Personale
FROM User1, User2;

I permessi che possono essere concessi (o revocati) agli utenti sono indicati con le seguenti parole chiave che vanno specificate dopo Grant o Revoke:

ALTER **per aggiungere od eliminare colonne, oppure per modificare i tipi di dati**
DELETE **per eliminare righe dalle tabelle**
INDEX **per creare indici**
INSERT **per inserire nuove righe nelle tabelle**
SELECT **per ritrovare i dati dalle tabelle**
UPDATE **per cambiare i valori contenuti nelle tabelle**
ALL **per tutti i permessi precedenti**

I permessi con le opzioni SELECT e UPDATE, nei comandi Grant e Revoke, diventano più restrittivi specificando, tra parentesi tonde e separati con la virgola, i nomi delle colonne che l'utente può vedere o modificare.

Per concedere il diritto di modifica del livello e dello stipendio base dei dipendenti all'utente denominato con User3, si deve usare il comando Grant nella forma:

GRANT UPDATE (Livello, StipBase)
ON Personale
TO User3;

Le viste

Il linguaggio SQL consente di decidere le modalità con le quali gli utenti possono vedere le tabelle del database, creando una finestra, detta VIEW (vista), su alcuni o su tutti i dati contenuti in una o più tabelle.

La vista viene indentificata con un nome assegnato in fase di creazione con il comando **CREATE VIEW.**

Con le viste è possibile decidere anche che un utente debba avere una visione solo parziale dei dati registrati nel database, mentre non può vedere altri dati: in questo modo si elimina il rischio di modifiche indesiderate su dati che non riguardano il lavoro di un determinato utente e nello stesso tempo l'utente non rimane distratto dai dati che sono ininfluenti sul tipo di elaborazione che deve fare.

L'utente può comunque operare sulla vista con i comandi illustrati nei paragrafi precedenti, come se fosse una tabella del database. ***Le viste sono finestre dinamiche sulle tabelle del database***, in quanto ogni modifica ai dati sulla tabella primaria è disponibile per l'utente attraverso la vista. Vale anche l'opposto: ogni modifica, sui dati della vista si riflette sui dati della tabella primaria. Non tutte le viste sono, però, aggiornabili: infatti le viste che contengono risultati di funzioni di aggregazione non possono essere modificate.

La creazione della vista viene realizzata con l'uso dell'istruzione ***SELECT*** all'interno del comando **CREATE VIEW.** Per creare una vista di nome ***Impieg*** contenente i dati dei dipendenti con funzione di Docente, si usa il comando:

```
CREATE VIEW Impieg
    AS SELECT *
        FROM   Personale
        WHERE   Funzione='Docente';
```

Una vista può essere eliminata con il comando **DROP VIEW** seguito dal nome assegnato alla vista in fase di creazione.

Per esempio il comando seguente elimina la vista creata con il nome Impieg:

```
DROP VIEW Impieg;
```

Il creatore della vista, usando il comando Grant illustrato in precedenza, fornisce all'utente il diritto di accesso ai dati attraverso l'utilizzo della vista.

Con il seguente esempio viene concesso all'utente denominato User3 il diritto di accedere alla vista Impieg, precedentemente creata, per ritrovare, cioè leggere, i dati in esso contenuti:

```
GRANT SELECT
  ON Impieg
  TO User3;
```

Esempi con uso del linguaggio SQL

Rappresentanti e fatture ai clienti

Testo del problema

Si vogliono organizzare le informazioni relative ai clienti, ai loro rappresentanti e alle fatture emesse. Ogni rappresentante ha tanti clienti, mentre a un cliente corrisponde un solo rappresentante. Per semplicità si supponga di registrare sola la data e l'importo totale di ogni fattura, senza specificare le righe di dettaglio sugli articoli venduti.

Definire il modello del database e rappresentare le seguenti interrogazioni:

1. Elenco dei clienti di un rappresentante.
2. Numero dei clienti affidati a un determinato rappresentate.
3. Cognome e nome del rappresentante di un determinato cliente.
4. Elenco delle fatture di un codice cliente prefissato, in ordine di data.
5. Fatturato raggruppato per codice rappresentante.
6. Numero delle fatture emesse raggruppate per codice rappresentante.
7. Nome del cliente e data di emissione per la fattura con importo prefissato.

Analisi dei dati

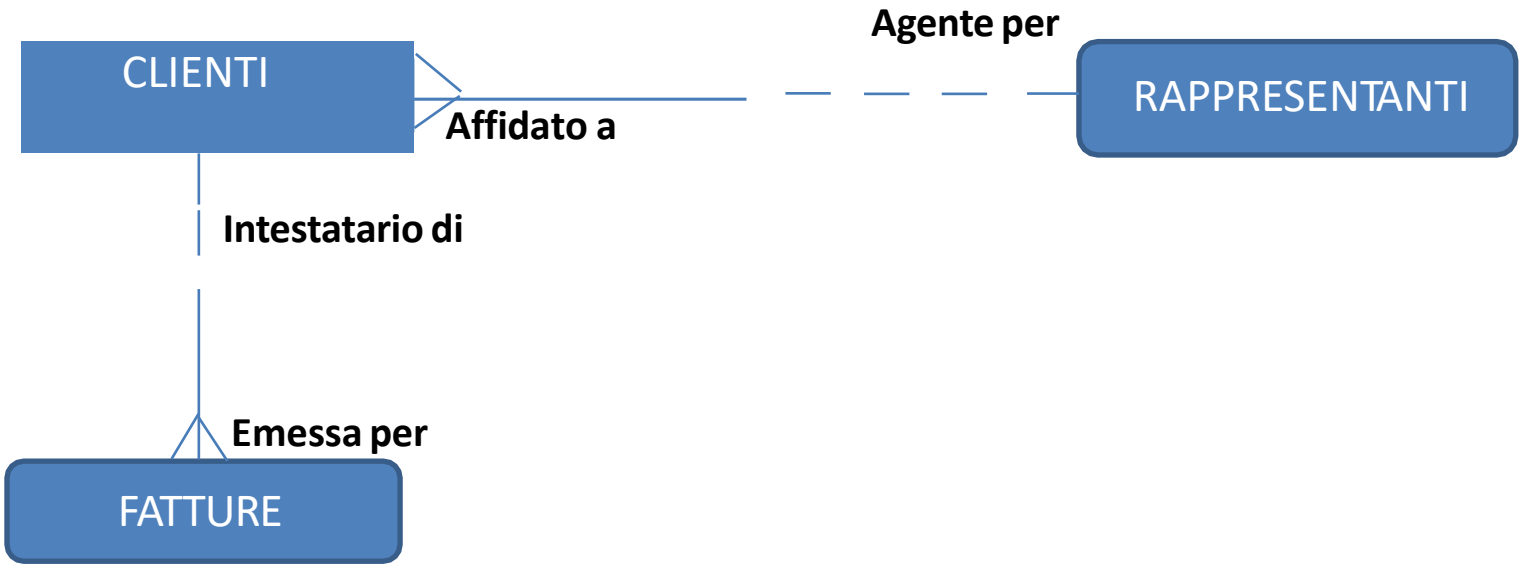
Le entità che possono essere individuate nel problema sono:

- **Cliente**, per i clienti ai quali vengono emesse le fatture;
- **Rappresentante**, per gli agenti ai quali sono assegnati i clienti;
- **Fattura,** per le vendite effettuate dai rappresentati.

Rappresentanti e fatture ai clienti

Gli attributi di **Clienti** sono: codice del cliente, ragione sociale, indirizzo, telefono, partita iva.
Gli attributi di **Rappresentanti** sono: codice del rappresentante, cognome, nome, zona assegnata.
Gli attributi di **Fatture** sono: numero della fattura, data di emissione, importo.
Tra l'entità **Rappresentanti** e l'entità **Clienti** si stabilisce un'associazione uno a molti perché ogni rappresentante ha tanti clienti, mentre a un cliente corrisponde un solo rappresentante.
Tra l'entità **Clienti** e l'entità **Fatture** esiste un'associazione uno a molti, perché per lo stesso cliente possono essere emesse molte fatture, ma ciascuna fattura è intestata a un solo cliente.
I dati del problema considerato si rappresentano graficamente con il seguente **schema E/R** (Entity/Relationship):

Modello E/R



Il modello viene verificato utilizzando le regole di lettura del modello E/R:
Ogni rappresentate può essere agente per uno o più clienti, ogni cliente deve essere affidato a un solo rappresentante. Ogni cliente può essere l'intestatario di una o più fatture, ogni fattura deve essere emessa per un solo cliente. Utilizzando le regole di derivazione si passa dal modello concettuale alle tabelle, introducendo le chiavi esterne per rappresentare le associazioni uno a molti.

Rappresentanti e fatture ai clienti

Tabelle

Rappresentanti (CodiceRap , CognomeRap, NomeRap, Zona)

Clienti (CodiceCli, RagioneSociale, Indirizzo, Telefono, PartitaIVA, *CodiceRap*)

Fatture(NumeroFatt , DataFatt, Importo, *CodiceCli*)

Le caratteristiche degli attributi delle tabelle sono illustrate nello schema seguente:

Tabella	Campo	Chiave	Formato	Dimensione
Rappresentanti	CodiceRap	Primaria	Carattere	4
	CognomeRap		Carattere	30
	NomeRap		Carattere	20
	Zona		Numerico	4
Clienti	CodiceCli	Primaria	Carattere	5
	RagioneSociale		Carattere	40
	Indirizzo		Carattere	25
	Telefono		Carattere	12
	PartitaIVA		Carattere	11
	CodiceRap	Esterna	Carattere	4
Fatture	NumeroFatt	Primaria	Numerico	5
	DataFatt		Data/ora	8
	Importo		Numerico	12
	CodiceCli	Esterna	Carattere	5

Interrogazioni

(Le costanti utilizzate nelle condizioni di ricerca sono indicate tra parentesi quadre)

1) Elenco dei clienti di un rappresentate.

Congiunzione di
(Selezione di Rappresentanti per CognomeRap=[rappresentante da controllare])
su CodiceRap e di Clienti su CodiceRap

```
SELECT RagioneSociale, Indirizzo , Telefono, PartitaIVA
  FROM Clienti, Rappresentanti
   WHERE Clienti.CodiceRap = Rappresentanti.CodiceRap
      AND CognomeRap=[rappresentante da controllare];
```

2) Numero dei clienti affidati a un determinato rappresentate.

Conteggio di
(Congiunzione di
(Selezione di Rappresentanti per CognomeRap=[rappresentante da controllare])
su CodiceRap e di Clienti su CodiceRap)

```
SELECT COUNT (*)
  FROM Clienti, Rappresentanti
  WHERE Clienti.CodiceRap=Rappresentanti.CodiceRap
      AND CognomeRap=[rappresentante da controllare];
```

Rappresentanti e fatture ai clienti

Interrogazioni

3) Cognome e nome del rappresentante di un determinato cliente.

Proiezione di
(Congiunzione di
(Selezione di Clienti per RagioneSociale=[cliente da controllare])
su CodiceRap e di Rappresentanti su CodiceRap)
Su CognomeRap, NomeRap

```
SELECT CognomeRap,NomeRap
  FROM Clienti, Rappresentanti
  WHERE Clienti.CodiceRap=Rappresentanti.CodiceRap
   AND RagioneSociale=[cliente da controllare];
```

4) Elenco delle fatture di un codice cliente prefissato, in ordine di data.

Selezione di Fatture per CodiceCli=[codice da controllare]

```
SELECT *
    FROM Fatture
    WHERE CodiceCli=[codice da controllare]
    ORDER BY DataFatt;
```

5) Fatturato raggruppato per codice rappresentante.

Proiezione di
((Congiunzione di Clienti su CodiceCli
e di Fatture su CodiceCli)
raggruppato per CodiceRap)
su CodiceRap e somma degli importi

```
SELECT CodiceRap, SUM (Importo)
FROM Fatture, Clienti
  WHERE Fatture.CodiceCli = Clienti.CodiceCli
  GROUP BY CodiceRap;
```

Interrogazioni

6) Numero delle fatture emesse raggruppate per codice rappresentante.

Proiezione di
((congiunzione di Clienti su CodiceCli)
e di Fatture su CodiceCli)
raggruppato per CodiceRap)
su CodiceRap e conteggio

```
SELECT CodiceRap, COUNT (*)
   FROM  Fatture, Clienti
   WHERE  Fatture.CodiceCli=Clienti.CodiceCli
   GROUP BY CodiceRap;
```

7) Nome del cliente e data di emissione per la fattura con importo prefissato.

Proiezione di
(Congiunzione di
(Selezione di Fattura per Importo=[quale importo])
su CodiceCli e di Clienti su CodiceCli)
su RagioneSociale, DataFatt, Importo

```
SELECT RagioneSociale, DataFatt, Importo
   FROM Fatture, Clienti
   WHERE Fatture.CodiceCli=Clienti.CodiceCli
     AND   Importo=[quale importo];
```

MICROSOFT ACCESS

Caratteristiche generali

Nelle lezioni precedenti sono stati descritti gli aspetti più significativi dei prodotti software per la gestione delle basi di dati e i comandi e le funzioni del linguaggio SQL. In queste unità didattiche viene presentato, a titolo di esempio, l'ambiente software **Microsoft Access**. Questo prodotto consiste principalmente in un sistema di gestione di basi di dati relazionali (**RDBMS**, Relational Database Management System) , può essere utilizzato su personal computer con sistema operativo Windows e permette di definire al suo interno applicazioni SQL.

In un database gestito da Access possono essere definite sette categorie di oggetti diversi:

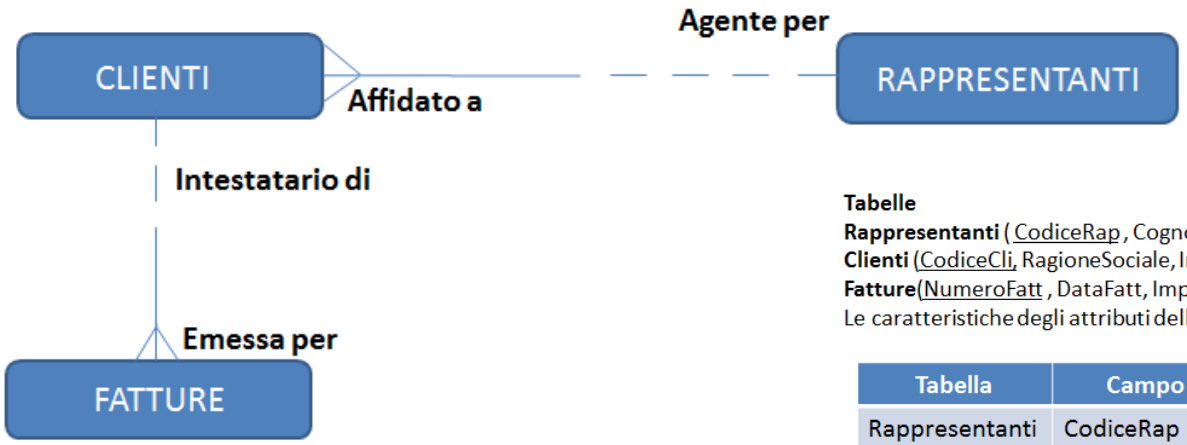
- Le **tabelle** costituiscono lo schema secondo cui sono organizzati i dati all'interno del database; è buona regola rispettare le forme di normalizzazione nella definizione delle tabelle, i dati raccolti nelle strutture tabellari formano il livello logico del database.
- Le **query** (interrogazioni sui dati) permettono di ricavare nuove tabelle dalle tabelle iniziali definite nel livello logico e formano, pertanto , il livello esterno del database; le tabelle ottenute attraverso query, generalmente , non rispettano le forme di normalizzazione.
- Le **maschere** e i **report** consentono di presentare all'utente finale i dati in modo da facilitare la loro consultazione (report) e il loro aggiornamento (maschere).
- Le **macro** e i **moduli** consentono di automatizzare operazioni particolari sui dati. In particolare le macro sono sequenze di comandi di Access raccolte in un'unica operazione che libera l'utente da lavori ripetitivi e noiosi, mentre i moduli sono procedure scritte in linguaggio interno ad Access, che permettono di eseguire operazioni più complesse di quelle che è possibile definire attraverso le macro.
- Le **pagine di accesso** sono utilizzate per visualizzare e pubblicare i dati del database su Internet utilizzando maschere generate automaticamente.

La definizione e l'apertura di un database

Per cominciare a conoscere alcune delle caratteristiche più significative di Access, consideriamo il problema esposto nelle pagine precedenti riguardante l'organizzazione delle informazioni relative ai clienti, ai loro rappresentanti e alle fatture emesse.

I dati sono rappresentati graficamente dal **modello E/R** sotto. Successivamente, utilizzando le regole di derivazione, si passa dal modello concettuale alle **tabelle** indicate nella parte bassa di destra del presente foglio.

Modello E/R



Rappresentanti e fatture ai clienti

Tabelle

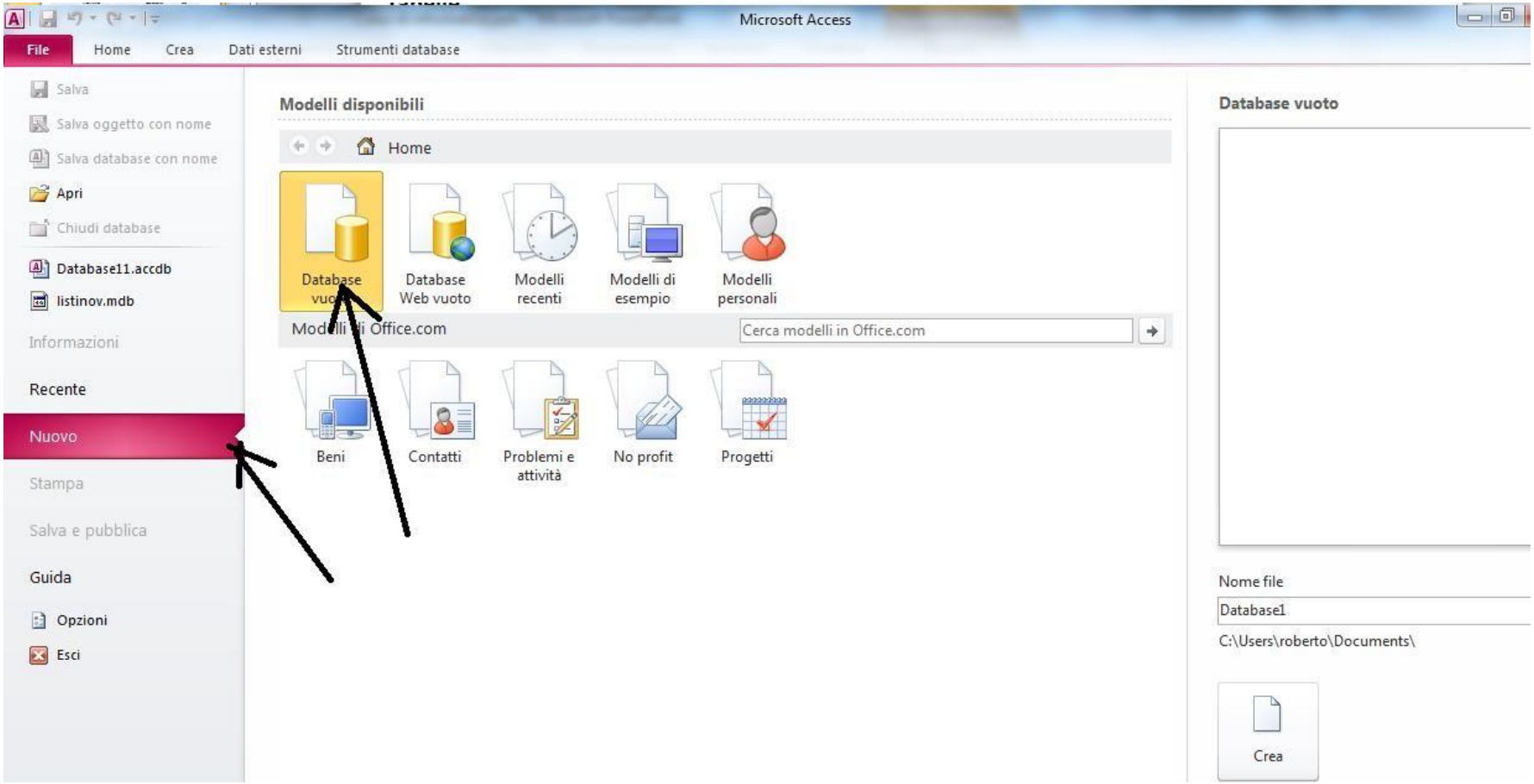
Rappresentanti (CodiceRap , CognomeRap, NomeRap, Zona)
Clienti (CodiceCli, RagioneSociale, Indirizzo, Telefono, PartitaIVA, *CodiceRap*)
Fatture(NumeroFatt , DataFatt, Importo, *CodiceCli*)
Le caratteristiche degli attributi delle tabelle sono illustrate nello schema seguente:

Tabella	Campo	Chiave	Formato	Dimensione
Rappresentanti	CodiceRap	Primaria	Carattere	4
	CognomeRap		Carattere	30
	NomeRap		Carattere	20
	Zona		Numerico	4
Clienti	CodiceCli	Primaria	Carattere	5
	RagioneSociale		Carattere	40
	Indirizzo		Carattere	25
	Telefono		Carattere	12
	PartitaIVA		Carattere	11
	CodiceRap	Esterna	Carattere	4
Fatture	NumeroFatt	Primaria	Numerico	5
	DataFatt		Data/ora	8
	Importo		Numerico	12
	CodiceCli	Esterna	Carattere	5

MICROSOFT ACCESS

La definizione e l'apertura di un database

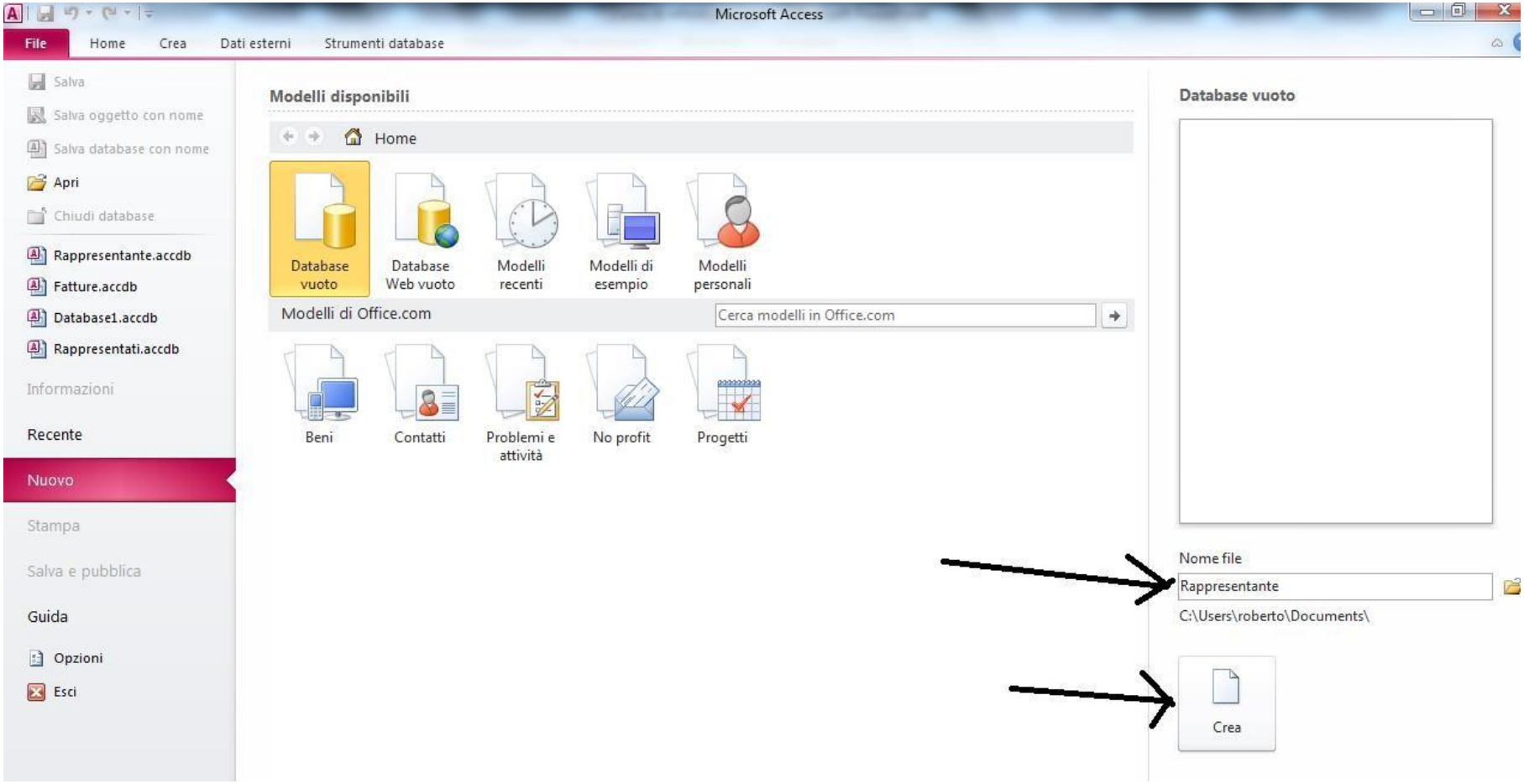
Access può essere attivato dal desktop di Windows cliccando sull'icona di **Avvio** e scegliendo **Tutti i Programmi**, e, di seguito, selezionare **Microsoft Office** - **Microsoft Access**. Inizialmente si presenta all'utente una finestra che chiede di indicare se si tratta di un nuovo database, se si vuole seguire la creazione guidata di un database o se si vuole aprire un database già esistente. Si scelga **Database vuoto**



MICROSOFT ACCESS

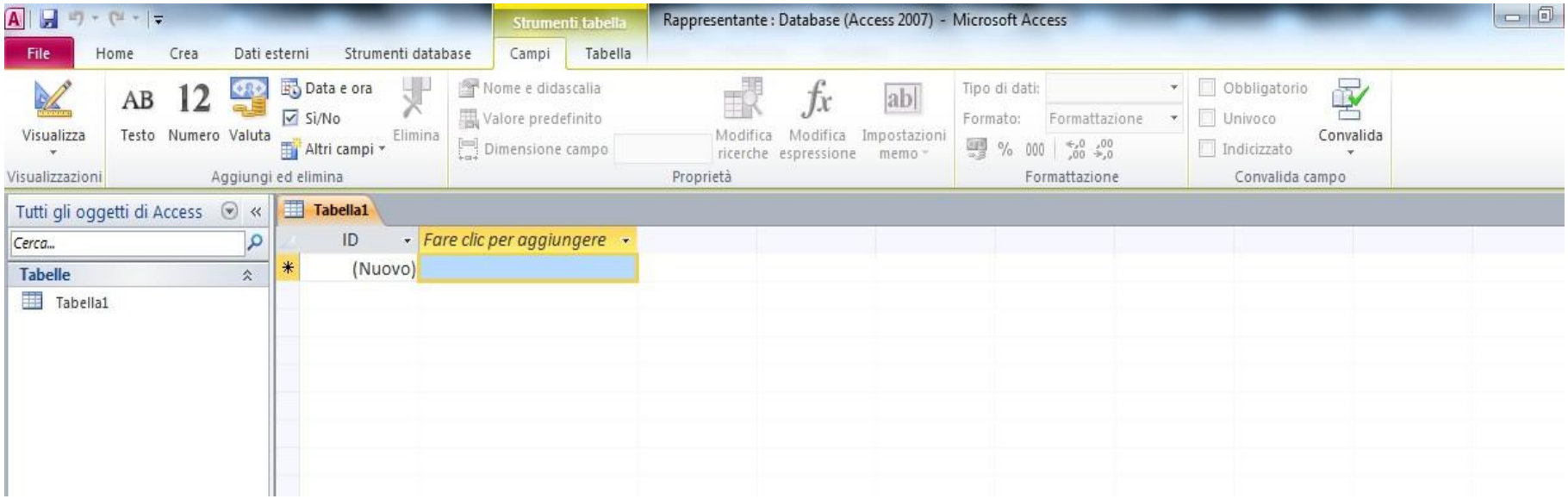
La definizione e l'apertura di un database

Nello spazio contrassegnato dal Nome File, area evidenziata dalla freccia, inserire il nome **Rappresentante** : questo è il nome che vogliamo dare al nostro nuovo database . Infine cliccare sull'icona **Crea.**



La definizione e l'apertura di un database

La finestra che si apre subito dopo aver cliccato su Crea è il vero e proprio ambiente di lavoro di Access.

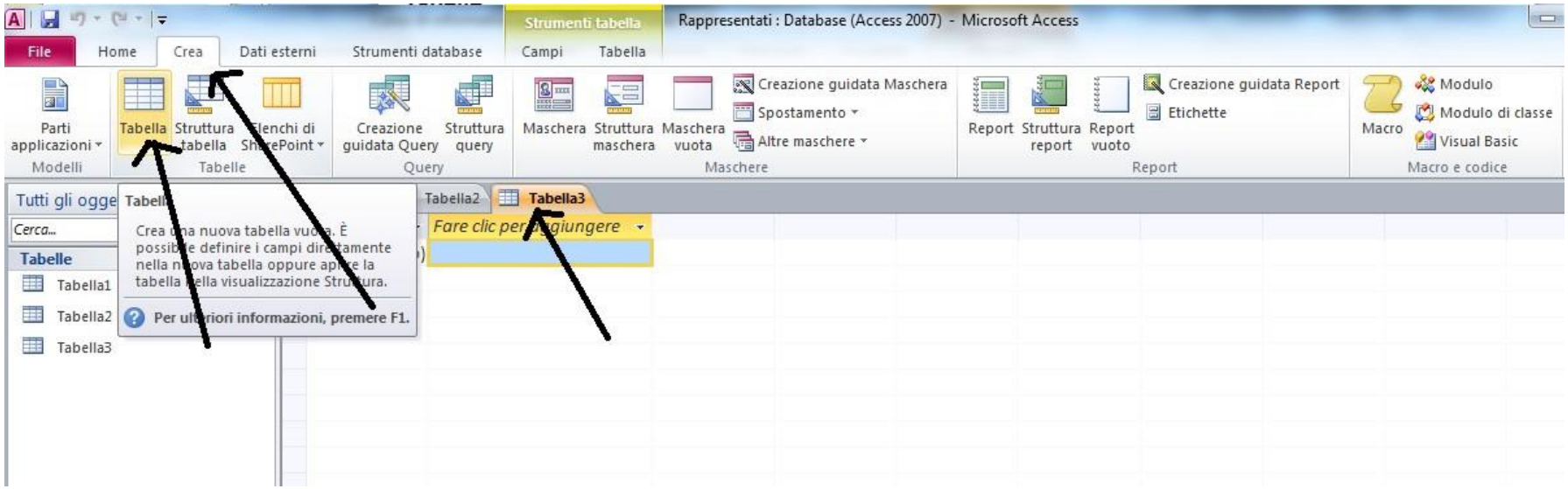


Prima di procedere ad illustrare che cosa si intenda per struttura di una tabella e come si definisca tale struttura vediamo come aprire un database. Nel caso in esame un database è già stato creato: anche se privo di dati e di oggetti, Access ha archiviato il nome del database Rappresentanti. Se a questo punto il lavoro venisse interrotto, nella successiva sessione di lavoro, per richiamare il database da completare, si deve procedere alla sua **apertura.**

Si attivi Access come visto in precedenza, ma questa volta si scelga nell'opzione **File**, quindi si selezioni la voce Rappresentanti.accdb per aprire il database creato in precedenza, ed infine la voce **Tabelle** della scheda **Crea** per tornare alla videata descritta in alto. Di default un database nuovo ha una sola tabella vuota, il cui nome è indentificato con Tabella1, ma se si clicca più volte su **Crea** - **Tabelle** verranno create più tabelle vuote.

La definizione e l'apertura di un database

Di default un database nuovo ha una sola tabella vuota, il cui nome è indentificato con Tabella1, ma se si clicca più volte su **Crea** - **Tabella** verranno create più tabelle vuote.



Tuttavia, se chiudiamo Access senza confermare il nome delle tabelle vuote create, l'ambiente non salverà i nuovi oggetti.

La definizione delle tabelle

Una tabella è un insieme di dati relativi ad una stessa entità. I dati di una tabella vengono rappresentati suddivisi in colonne e righe. Spesso si ricorre alla terminologia informatica tradizionale: le colonne vengono denominati **campi** e le righe **record** .

Per definizione di tabella si intende la descrizione della struttura della tabella specificando per ogni colonna:

- Il nome della colonna;
- Il tipo di dati che essa contiene;
- La dimensione massima (in byte) prevista per i dati da inserire nella colonna.

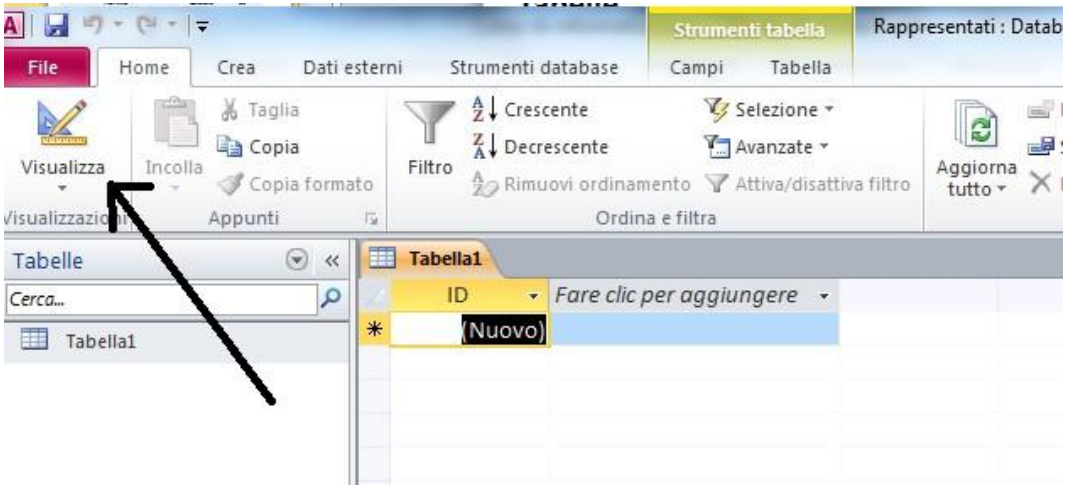
La definizione e l'apertura di un database

Si procede ora ad applicare al problema in esame la nozioni viste in precedenza.

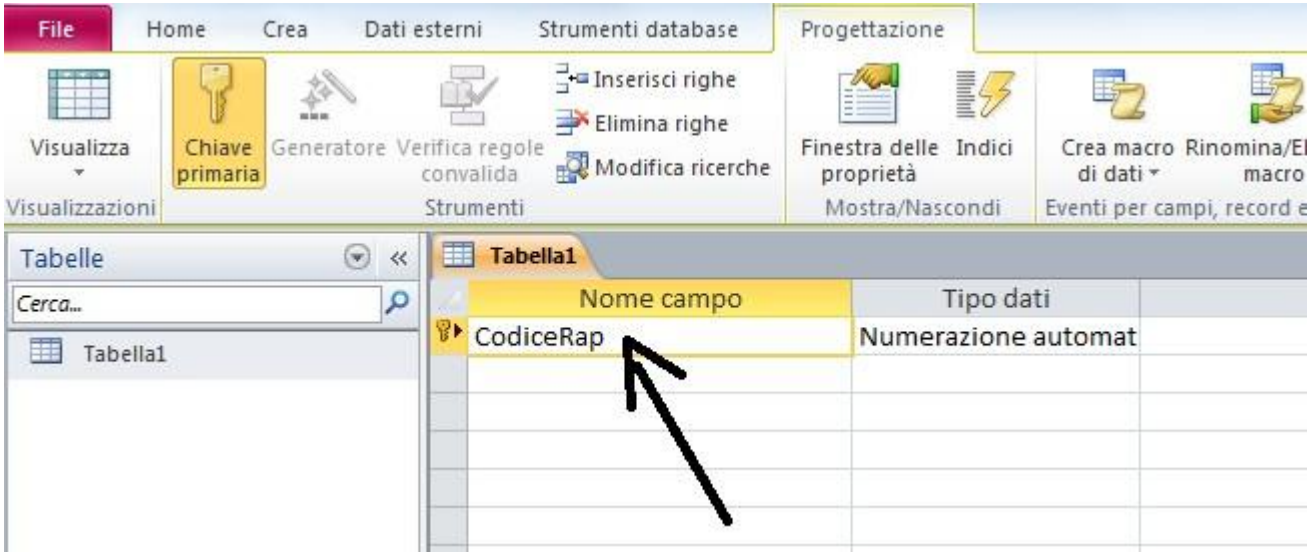
Con riferimento ai dati del problema esposto, si deve definire, per ciascuna colonna il nome, il tipo e la dimensione dei dati contenuti.

Creazione della tabella Rappresentanti:

Supponendo di aver già aperto Access, si faccia clic sulla voce Visualizza (vedi figura a lato).

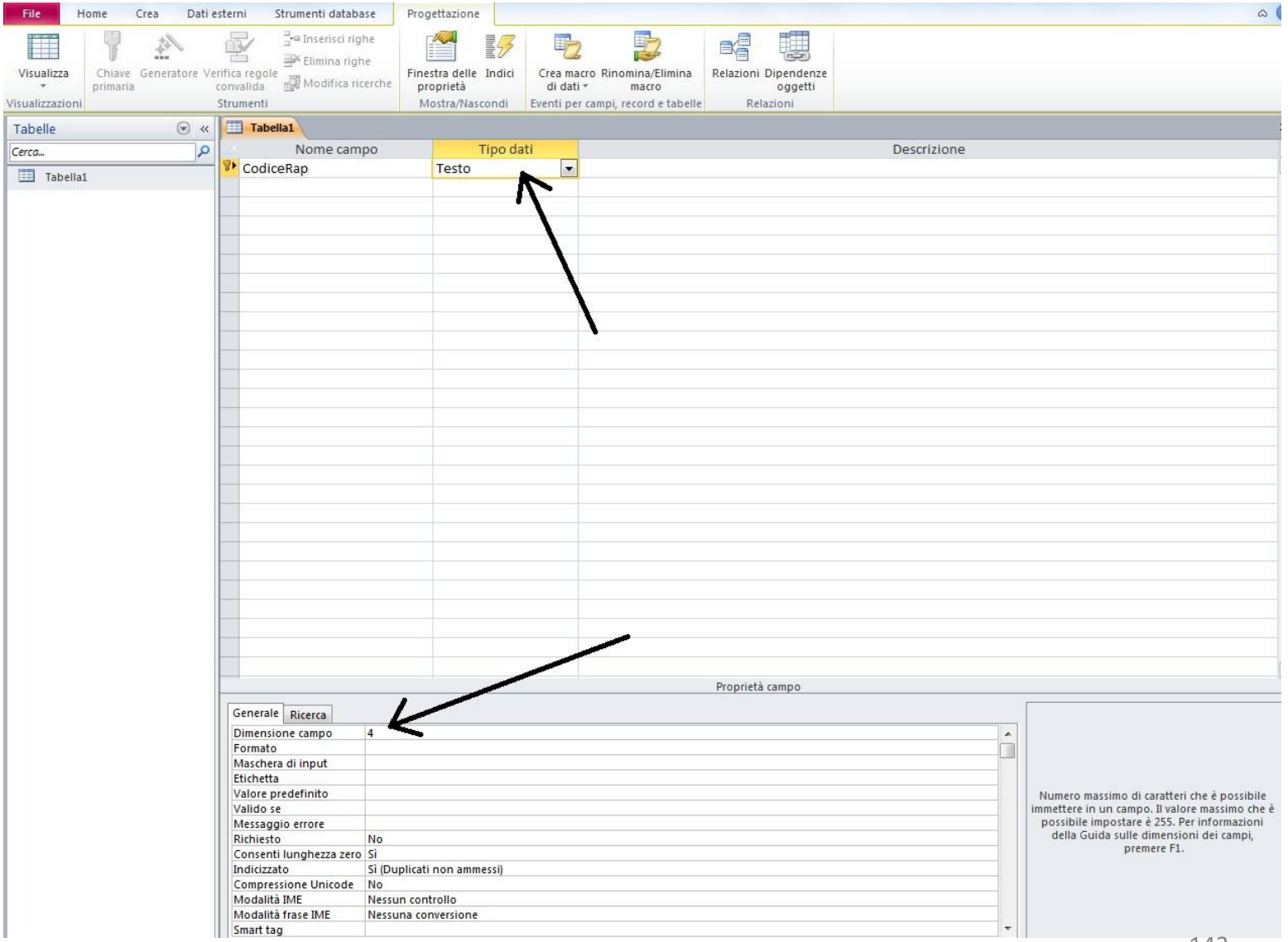


Quindi cliccare su **Nome campo** e digitare CodiceRap (figura in basso).



La definizione e l'apertura di un database

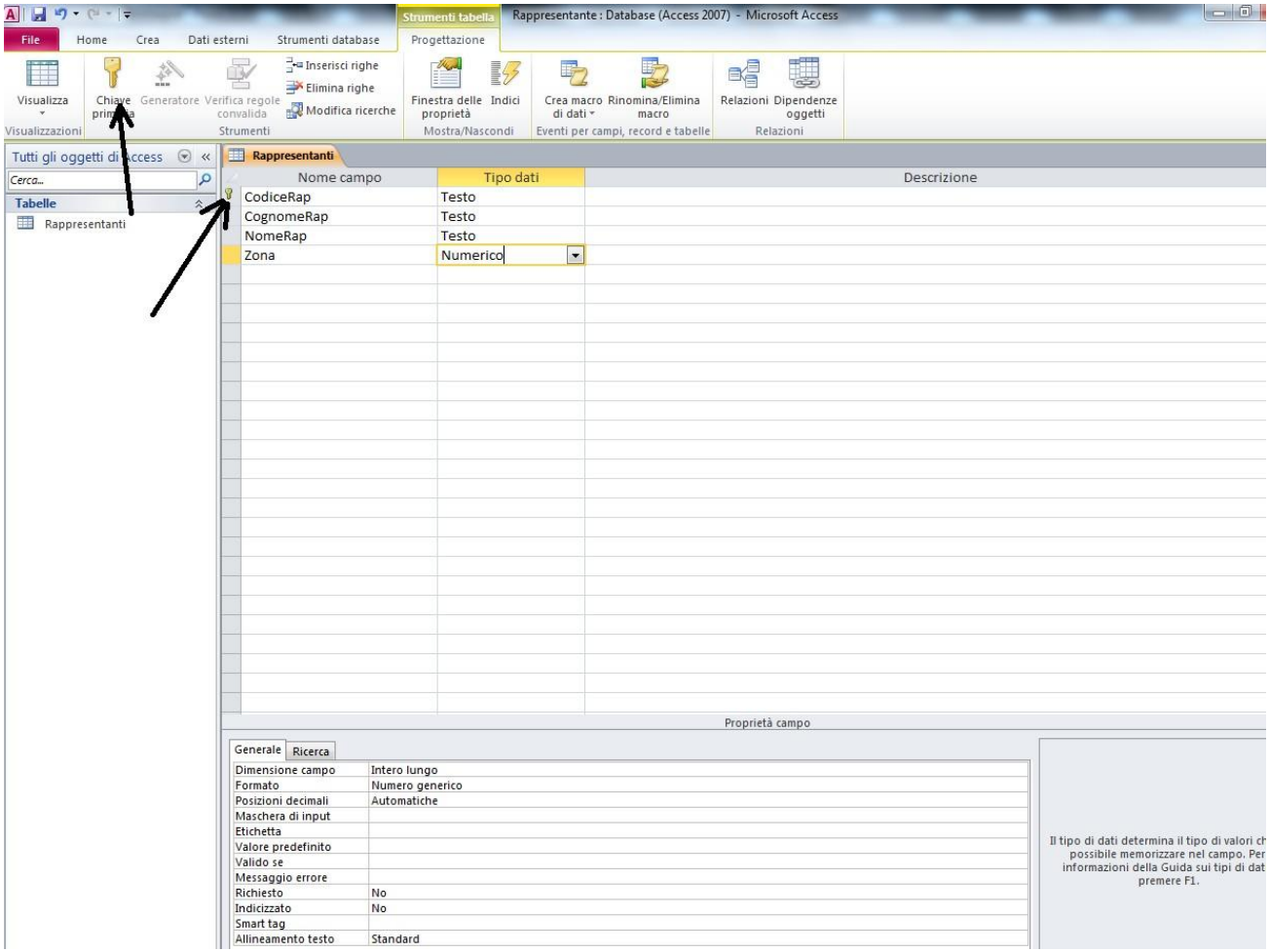
Inserire la voce **Testo** nella casella in corrispondenza di **Tipo dati** ed il numero **4 nella casella Dimensione Campo,** così come indicato nella figura sotto: In questo modo vogliamo indicare che **CodiceRap** è un **testo** di **4** caratteri.



La definizione e l'apertura di un database

Completare con la definizione degli altri campi della tabella **Rappresentanti,** così come riportato in figura. Prima di passare alla creazione delle altre tabelle , si consiglia di verificare se è stata associata la chiave primaria **(primary key)** al campo CodiceRap.

Ricordiamo che quando si specifica un vincolo **PRIMARY KEY** per una tabella, il motore del database assicura l'univocità dei dati creando un indice univoco per le colonne chiave primaria. Questo indice consente inoltre di accedere rapidamente ai dati quando si utilizza la chiave primaria nelle query. Le chiavi primarie scelte devono pertanto essere conformi alle regole stabilite per la creazione di indici univoci.



La definizione e l'apertura di un database

Con operazioni analoghe viene definita la struttura della tabella *Fatture* e *Clienti* con il risultato indicato nella figura sotto.

Clienti

Nome campo	Tipo dati
CodiceCli	Testo
RagioneSociale	Testo
Indirizzo	Testo
Telefono	Testo
ParitaIVA	Testo
CodiceRap	Testo

Fatture

Nome campo	Tipo dati
NumeroFatt	Numerico
DataFatt	Data/ora
Importo	Numerico
CodiceCli	Testo

Rappresentanti

Nome campo	Tipo dati
CodiceRap	Testo
CognomeRap	Testo
NomeRap	Testo
Zona	Numerico

La definizione delle associazioni

Uno degli aspetti più interessanti di Access è la possibilità di definire e di memorizzare le **associazioni** esistenti tra le tabelle del database su ci si sta lavorando. Inoltre, la definizione delle associazioni è realizzata utilizzando in modo molto intuitivo l'interfaccia grafica.

Riprendendo l'esempio presentato all'inizio di questo capitolo, nel ***modello E/R*** è indicato che tra le tabelle ***Rappresentanti*** e ***Clienti*** esiste una associazione **uno** a **molti** e che tale associazione è realizzata dalla chiave esterna ***CodiceRap*** presente nella tabella ***Clienti***

Tabella	Campo	Chiave	Formato	Dimensione
Rappresentanti	CodiceRap	Primaria	Carattere	4
	CognomeRap		Carattere	30
	NomeRap		Carattere	20
	Zona		Numerico	4

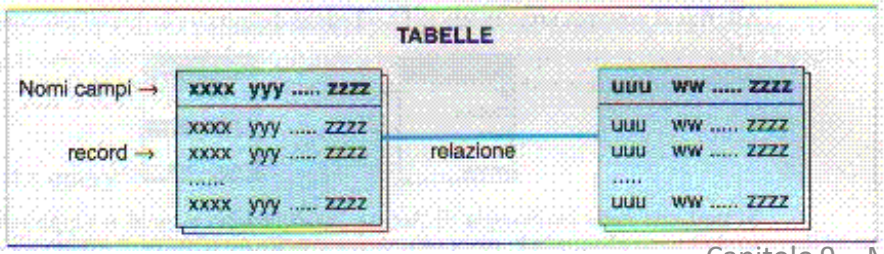
1

Tabella	Campo	Chiave	Formato	Dimensione
Clienti	CodiceCli	Primaria	Carattere	5
	RagioneSociale		Carattere	40
	Indirizzo		Carattere	25
	Telefono		Carattere	12
	PartitaIVA		Carattere	11
	CodiceRap	Esterna	Carattere	4

N

Prima di presentare la definizione delle associazioni in Access, è necessario precisare alcuni aspetti riguardanti la terminologia più diffusa con la definizione dei termini relazioni, tuple, domini e associazioni, il cui significato può essere sintetizzato dallo schema sotto.

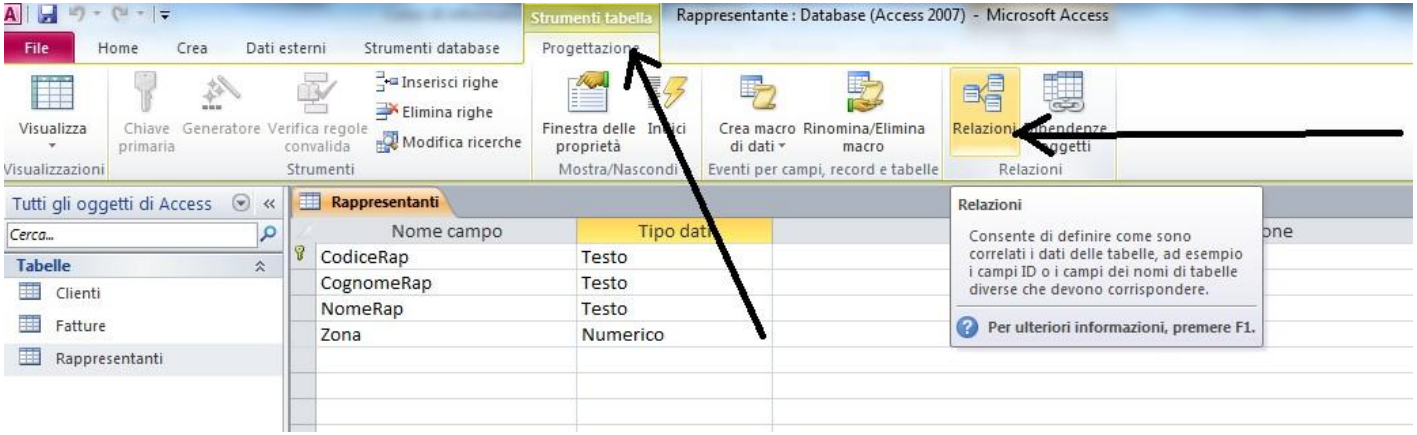
Access nella versione italiana, usa prevalentemente i termini tabelle, campi , record e relazioni. Il loro significato può essere sintetizzato dallo schema a destra. Esso è del tutto equivalente a quello sopra e mette in evidenza l'uso diverso che nei due casi viene fatto del medesimo termine *relazione*.



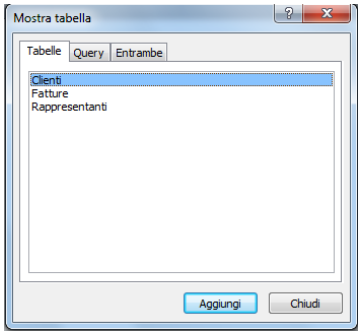
La definizione delle associazioni

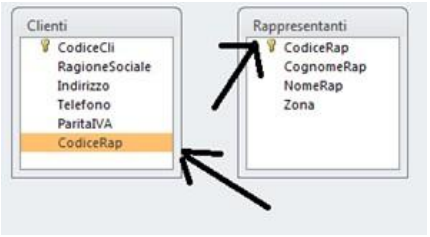
Viene descritta ora la procedura per la definizione di relazioni tra le tabelle di Access**.**

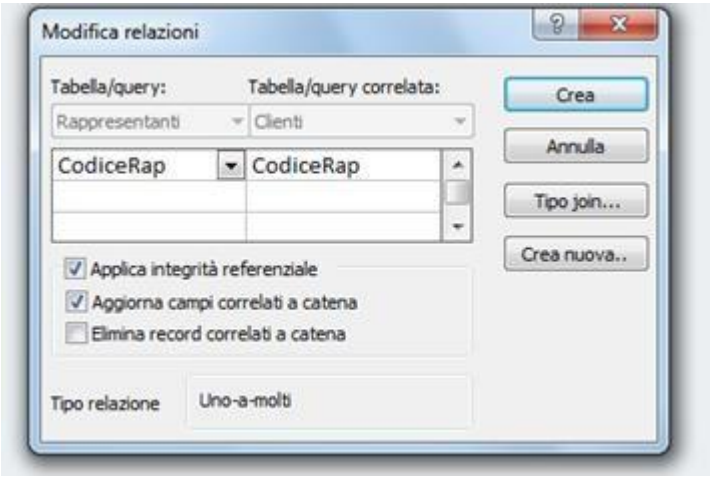
1. Il primo passo consiste nel fare un semplice clic sull'icona che rappresenta le **relazioni.** Quindi, selezionare la scheda Progettazione e nel gruppo Relazioni selezionare la voce Relazioni.



2. Si apre la finestra di dialogo Mostra tabella per specificare su quali oggetti (tabelle e/o query) si vogliono definire le relazioni.
3. Su ciascun nome di tabella si faccia clic per selezionarla, clic sul pulsante Aggiungi e, da ultimo, clic su **Chiudi.**
4. Per definire le relazioni tra Rappresentanti e Clienti si porti il mouse su CodiceRap della tabella Clienti, tenendo premuto il tasto sinistro, si trascini il mouse fino a sovrapporsi al campo CodiceRap di Rappresentanti: rilasciando il pulsante sinistro si apre una nuova finestra che indica che appunto è stata creata l'associazione tra le due tabelle attraverso il campo CodiceRap.
5. Nella nuova finestra compare anche la domanda Applica integrità referenziale. È opportuno attivare questo controllo facendo clic sulla casella di spunta a sinistra dell'opzione.





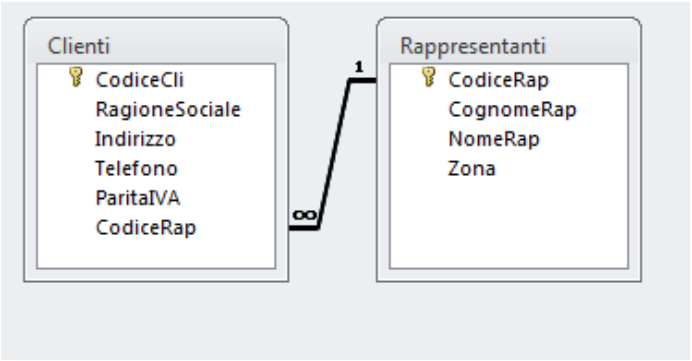


La definizione delle associazioni

L'integrità referenziale è applicabile:

1. Se il campo con cui si genera la relazione
   - è campo chiave di una delle due tabelle,
   - ha il medesimo tipo dati nelle due tabelle
2. Se le due tabelle risiedono nel medesimo database.

Alla fine si presenta uno schema della relazione definita tra le due tabelle come in figura sotto .

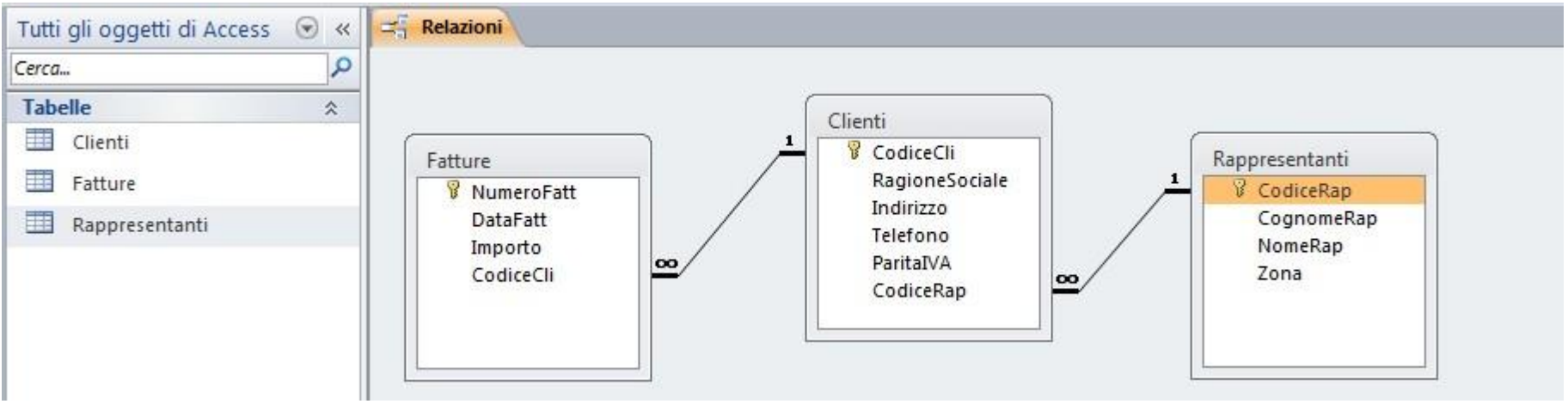


Si noti che nella simbologia di Access l'associazione a molti è rappresentata con il simbolo ∞ ed inoltre tutte le linee che rappresentano le associazioni sono continue : vale a dire che non esiste la distinzione tra verso obbligatorio e verso opzionale dell'associazione.

Nella finestra di dialogo Modifica relazioni (figura in alto a destra) oltre ad attivare l'opzione Applica integrità referenziale è possibile anche richiedere le funzioni Aggiorna campi correlati a catena e Elimina campi correlati a catena. L'aggiornamento dei campi correlati fa in modo che le modifiche sul campo CodiceRap nella tabella Rappresentanti comporti anche l'aggiornamento del campo CodiceRap nella tabella Clienti; mentre l'eliminazione dei campi correlati fa in modo che l'eliminazione di una riga della tabella Rappresentanti determini l'eliminazione dalla tabella Clienti di tutte quelle righe che avevano il campo CodiceRap uguale al CodiceRap della riga eliminata. Nel nostro caso non attiviamo l'eliminazione dei record correlati a catena.

La definizione delle associazioni

Concludiamo l'associazione esistente tra la tabella Clienti e la tabella Fatture, rispettando le regole descritte nello schema E/R e procedendo così come descritto nelle pagine precedenti.
Otteniamo uno schema delle relazioni definito nella figura sotto:



Salviamo il tutto e passiamo al popolamento delle varie tabelle. Per popolamento di tabelle si intende l'operazione di caricamento dei dati negli archivi del database di Access in modo tale da permettere successivamente le interrogazioni dei dati presenti o altre operazioni da parte dell'utente.
Partiamo con l'inserimento dei dati nella tabella **Rappresentanti** in quanto essa non dipendono dalle altre tabelle. Diversamente, per popolare la tabella Clienti bisogna prima decidere a quale CodiceRap, della tabella Rappresentanti, far riferimento. In secondo luogo, prima di popolare la tabella Fatture è opportuno riempire la tabella Clienti, in quanto non è possibile emettere nessuna fattura se prima non si crea il clienti destinatario della fattura.

Inserimento dati nelle tabelle

Rappresentanti

Rappresentanti

CodiceRap	CognomeRa	NomeRap	Zona	*Fare clic per aggiungere*
0024	Rossi	Marco	1	
0025	Verdi	Michele	2	
0026	Bianchi	Antonio	3	
0027	Cascardi	Maria	4	
0028	Della Valle	Antonio	5	
0029	Risi	Roberto	6	

Clienti

Rappresentanti | Clienti

CodiceCli	RagioneSoci	Indirizzo	Telefono	ParitaIVA	CodiceRap
AZ21	Ditta Start	Via XX Settembre	0776266786	02444523234	0024
AB23	Società Pallon	Via Verdi, 2	07764567523	01231231313	0025
AC35	Ditta F.lli Marc	Viale America, 44	062311334	01234242423	0026

Fatture

Rappresentanti | Clienti | Fatture

NumeroFatt	DataFatt	Importo	CodiceCli	*Fare clic per aggiungere*
1	01/10/2014	34000	AZ21	
2	02/10/2014	45000	AB23	
3	03/10/2014	56000	AB23	
4	04/10/2014	40000	AZ21	
*				

Creazione query

Per **query** si intende un'interrogazione che viene posta riguardo ai dati presenti in un database. I dati che forniscono la riposta alla domanda definita dalla **query** possono provenire da una o più tabelle; inoltre, i dati prodotti da una query sono organizzati secondo la struttura di una tabella.

Con riferimento al problema precedente, si riportano le 7 query richieste :

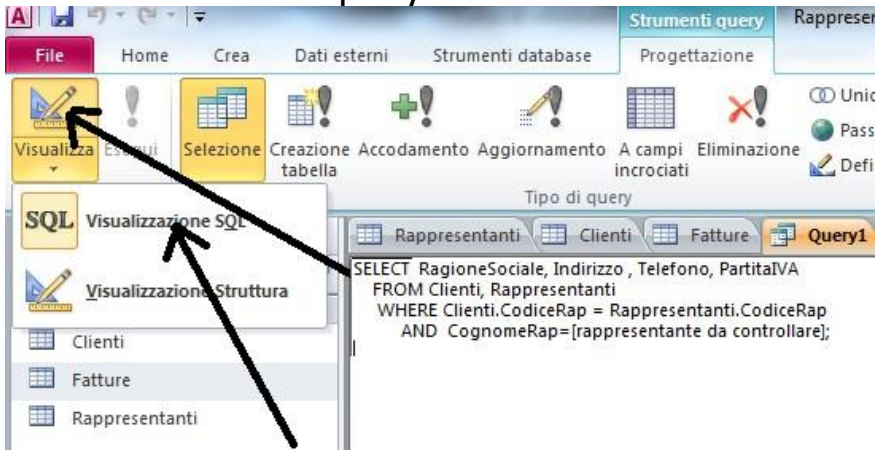
1. Elenco dei clienti di un rappresentante.
2. Numero dei clienti affidati a un determinato rappresentate.
3. Cognome e nome del rappresentante di un determinato cliente.
4. Elenco delle fatture di un codice cliente prefissato, in ordine di data.
5. Fatturato raggruppato per codice rappresentante.
6. Numero delle fatture emesse raggruppate per codice rappresentante.
7. Nome del cliente e data di emissione per la fattura con importo prefissato.

È possibile procedere alla definizione di una query attraverso lo strumento **QBE** (Query By Example, interrogazione attraverso un esempio). La struttura grafica della finestra QBE consente di utilizzare il mouse per selezionare, trascinare e gestire in vari modi gli oggetti in essa contenuti e per descrivere un esempio (da qui il nome) delle righe che andranno a comporre la tabella risultato della query.

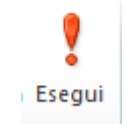
La procedura per la creazione di una query

Per inserire la nostra query procediamo nel seguente modo:

1) Selezionare la scheda **Crea;**
2) nel gruppo **Query,** cliccare su **Struttura Query** e successivamente fare clic **Chiudi;**
4) clicchiamo sull'icona Visualizza e SQL e a destra del riquadro inseriamo la nostra query che mostra l'elenco dei clienti di un determinato rappresentante, così come riportata nella figura.

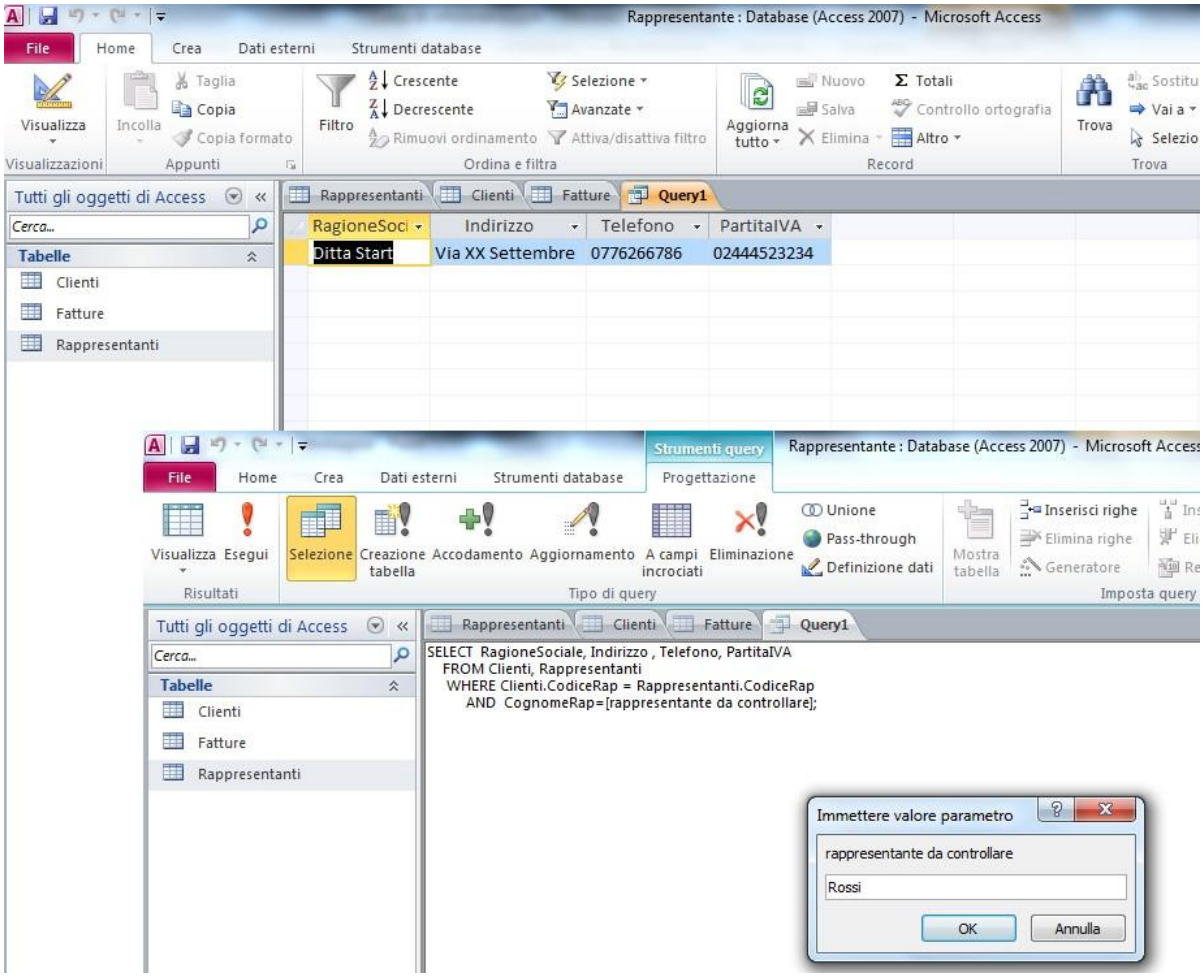


Creazione query

Per visualizzare il risultato di una query, cliccare sull'icona Esegui 

ed inserire il cognome del rappresentante da controllare, successivamente verranno visualizzati i cliente appartenenti a quel rappresentate. Per salvare la query ed assegnare alla stessa un nome mnemonico, che ne indichi la funzione, si consiglia di cliccare sulla scheda Query1 e con il tasto destro del mouse selezionare rinomina, infine assegnare il nome alla query. Ad esempio, assegniamo il nome «clienti_rap» alla query precedentemente creata.

Con operazioni analoghe sarà quindi possibile creare le altre Query .



Creazione di tutte le query

Riportiamo di seguito la query n. 4 ed i risultati ottenuti . Si precisa che nella fase di input è stato inserito il codice cliente AB23.

Rappresentanti | Clienti | Fatture | Clienti_rap | Num_rapp_clienti | ElencoFatt_cliente_ord_per_data

NumeroFatt	DataFatt	Importo	CodiceCli
2	02/10/2014	45000	AB23
3	03/10/2014	56000	AB23
*			

Codice della query «**Elenco delle fatture di un codice cliente prefissato, in ordine di data»:**

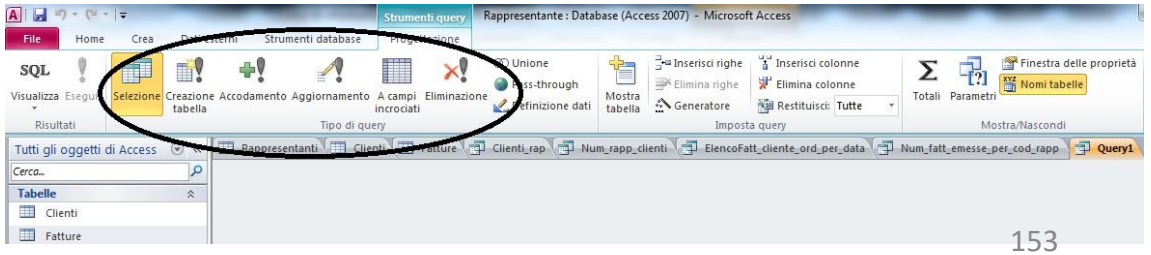
```
SELECT *
    FROM  Fatture
    WHERE CodiceCli=[codice da controllare]
    ORDER BY DataFatt;
```

Salviamo infine il database contente le 3 tabelle e le 7 query ciascuna rinominata il proprio nome mnemonico.

I tipi di query in Access

In Access esistono diversi categorie di query, ad esempio abbiamo le query di comando che apportano modifiche a molti record con una sola operazione. Esistono quattro tipi di query di comando: di creazione tabella, di aggiornamento, di accodamento e di eliminazione. Nella figura sotto vediamo come è possibile scegliere il tipo di query di comando.

Selezioniamo la scheda **Progettazione** ed il gruppo **Tipi di query,** quindi cliccare su una delle Icone per applicare una delle varie categorie di query .

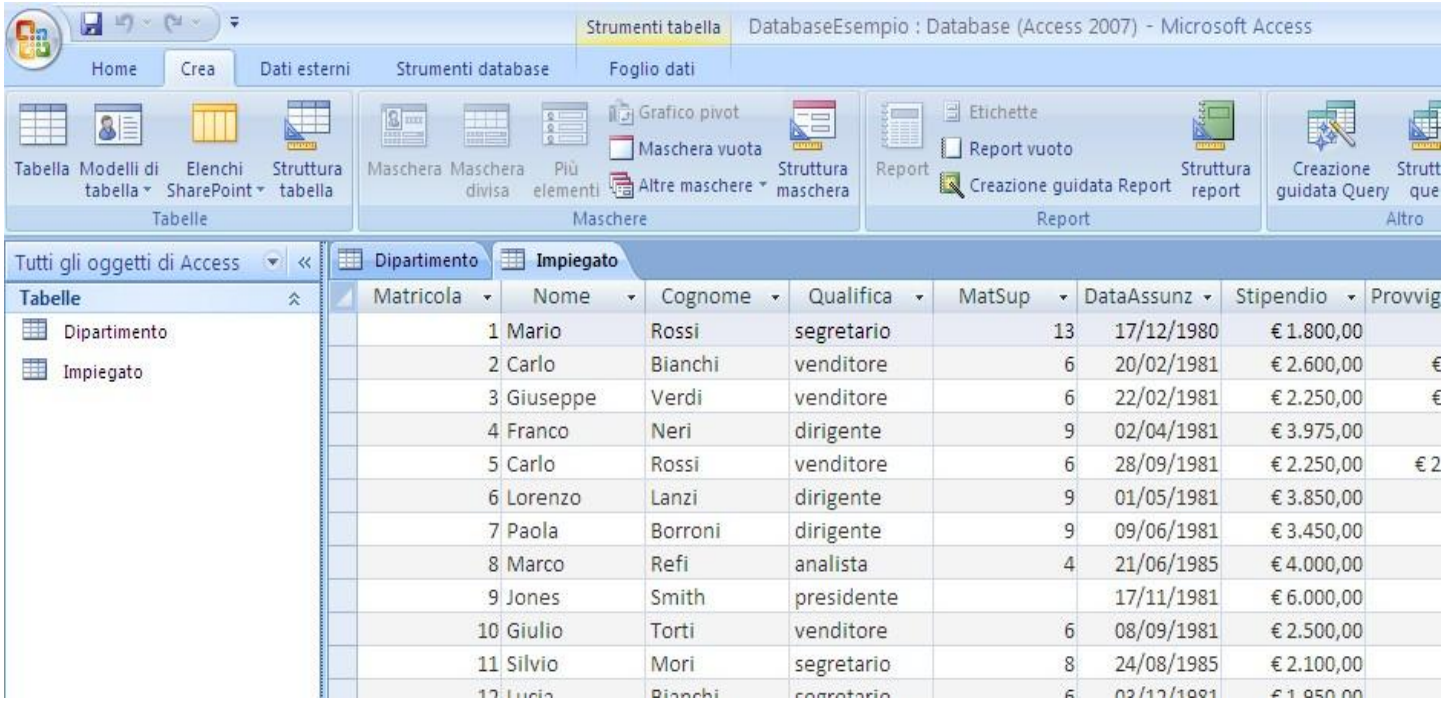


Le maschere di Access

Con la maschera di Access è possibile creare un'interfaccia grafica che consente un approccio più semplice ed intuitivo tra l'utente e la base di dati. Le maschere permettono di consultare ed inserire velocemente i dati di una tabella o visualizzare i dati di una query; possono contenere grafici, naturalmente caselle di testo che fanno riferimento a campi specifici di una data tabella, immagini, etichette con testo personalizzato, riquadri e addirittura anche altre maschere.

Creare una maschera

Come per la maggior parte degli oggetti che puoi creare in un database, hai più alternative tra cui scegliere. Per creare una maschera, per esempio, puoi scegliere tra le varie opzioni dalla scheda **Maschere**, selezionabile dal menu **Crea**. Al fine di essere esaustivi, utilizzeremo il database di esempio(figura sotto) contenente la tabella Dipartimento ed Impiegato.



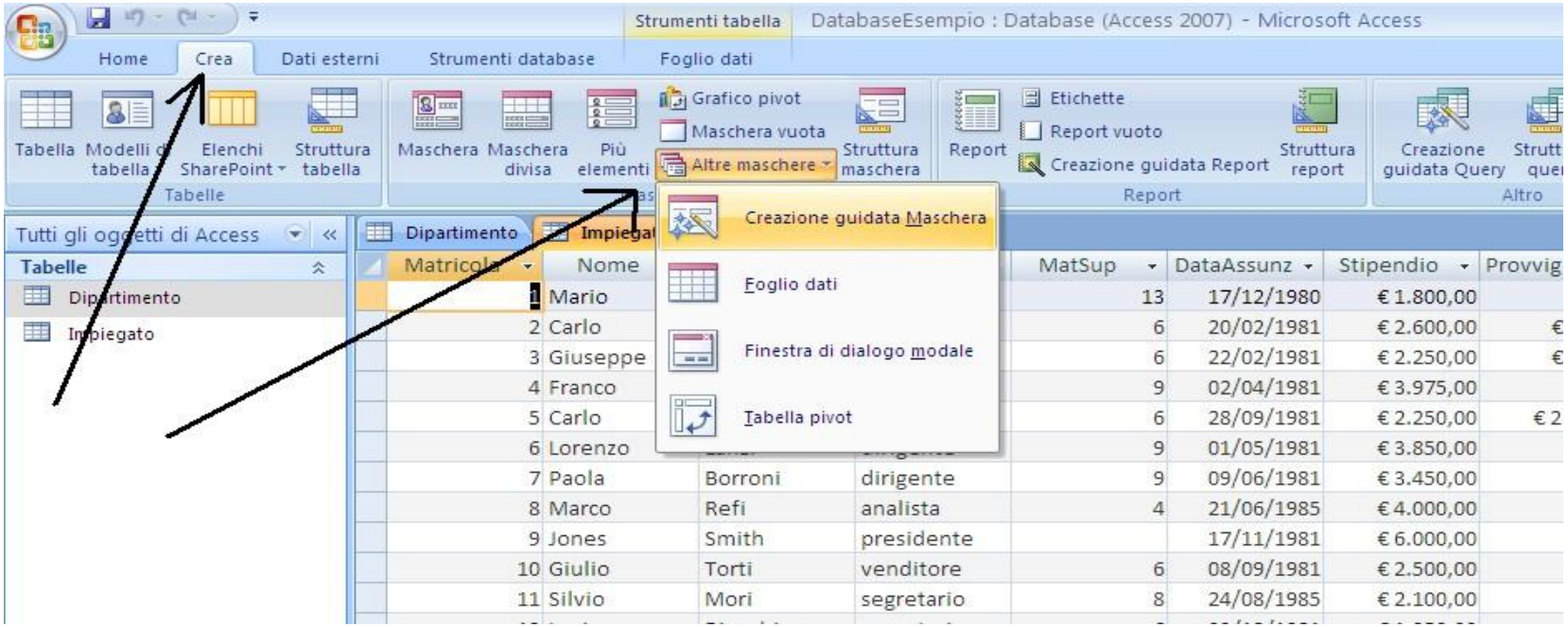
Matricola	Nome	Cognome	Qualifica	MatSup	DataAssunz	Stipendio	Provvig
1	Mario	Rossi	segretario	13	17/12/1980	€ 1.800,00	
2	Carlo	Bianchi	venditore	6	20/02/1981	€ 2.600,00	€
3	Giuseppe	Verdi	venditore	6	22/02/1981	€ 2.250,00	€
4	Franco	Neri	dirigente	9	02/04/1981	€ 3.975,00	
5	Carlo	Rossi	venditore	6	28/09/1981	€ 2.250,00	€ 2
6	Lorenzo	Lanzi	dirigente	9	01/05/1981	€ 3.850,00	
7	Paola	Borroni	dirigente	9	09/06/1981	€ 3.450,00	
8	Marco	Refi	analista	4	21/06/1985	€ 4.000,00	
9	Jones	Smith	presidente		17/11/1981	€ 6.000,00	
10	Giulio	Torti	venditore	6	08/09/1981	€ 2.500,00	
11	Silvio	Mori	segretario	8	24/08/1985	€ 2.100,00	

Le maschere di Access

Nel nostro esempio di creazione maschera, abbiamo scelto la **Creazione guidata Maschere** che crea con pochi passi un'utile interfaccia adatta per l'inserimento e la visualizzazione dei dati appartenenti alla tabella Impiegato. Con **Creazione guidata Maschera**, puoi specificare il tipo di maschera che vuoi creare; la procedura ti guida attraverso tutti i passaggi necessari. Rispondi alla serie di domande e Access creerà una maschera utilizzando le informazioni e le impostazioni che hai precedentemente inserito.

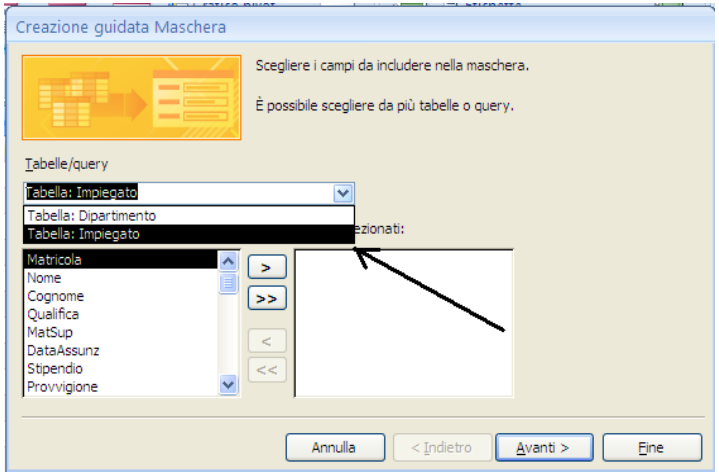
Di seguito verranno descritti i passi da seguire per creare una maschera personalizzata con **Creazione guidata Maschere**.

Dal menu **Crea,** per procedere con la personalizzazione della maschera**,** selezionare **Altre maschere,** quindi cliccare su **Creazione guidata Maschere** (vedi figura sotto).

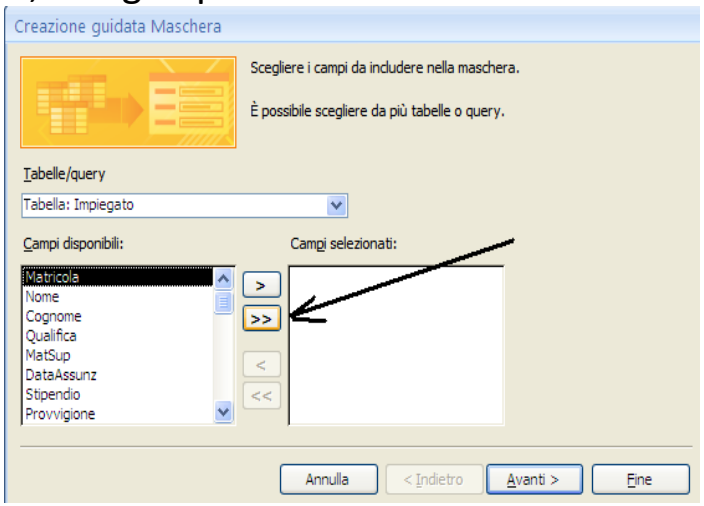


Le maschere di Access

La procedura predispone una finestra di dialogo, così come indicato nella figura sotto, dalla quale sarà possibile selezionare la tabella o la query su cui basare la maschera. Quindi selezioniamo la tabella Impiegato.

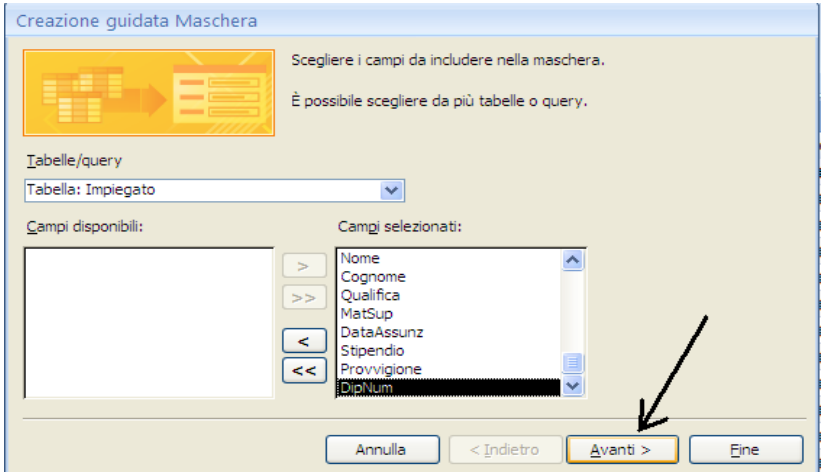


Dal riquadro **Campi disponibili**, selezionare i campi sui quali si vuole costruire la maschera: si può scegliere se selezionare un campo alla volta, premendo il pulsante con il simbolo **>**; diversamente, se vogliamo selezionare tutti i campi della tabella, bisogna premere sul simbolo **>>.** Nel nostro caso selezioniamo tutti i campi(vedi figura sotto)

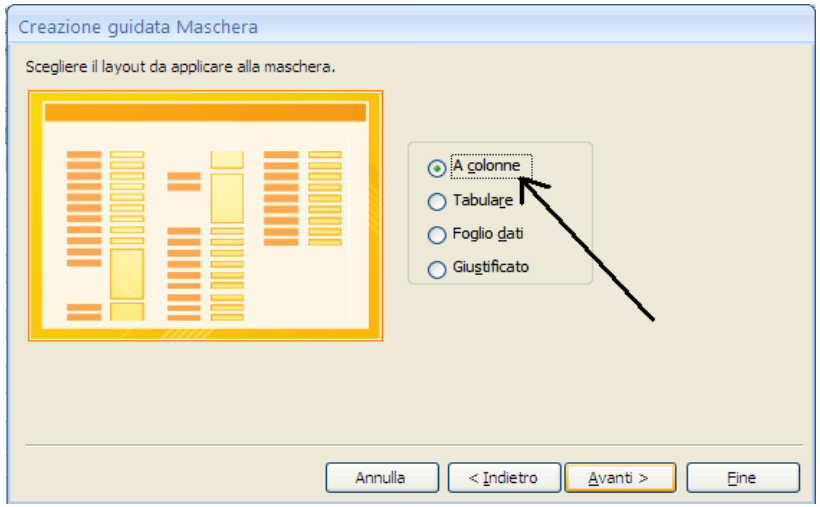


Le maschere di Access

Premiamo il pulsante **Avanti** (vedi figura).

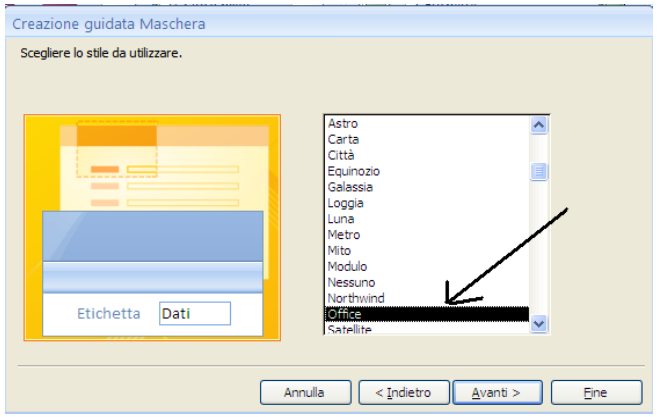


Decidiamo ora il layout delle informazioni sulla maschera (a colonna, Tabulare, Foglio dati, Giustificato, Tabella pivot, Grafico pivot). Nell'area di anteprima della finestra di dialogo, in alto a sinistra, possiamo vedere come apparirà la maschera in funzione della scelta. Nel nostro caso utilizziamo il layout a colonne, così come riportato nella figura successiva.

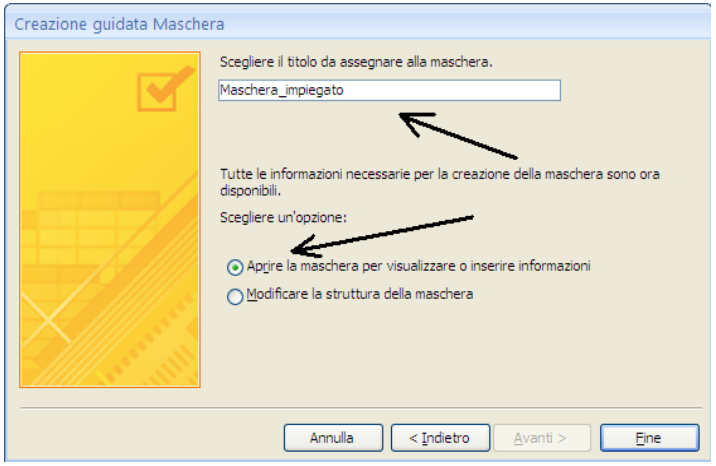


Le maschere di Access

Premere **Avanti** e specificare lo stile della maschera, che influisce sulla formattazione e sull'aspetto finale; anche in questa schermata è possibile vedere l'anteprima dello stile selezionato (parte sinistra della figura). Utilizziamo per il nostro esempio lo stile **Office**, quindi premiamo **Avanti** (figura sotto).

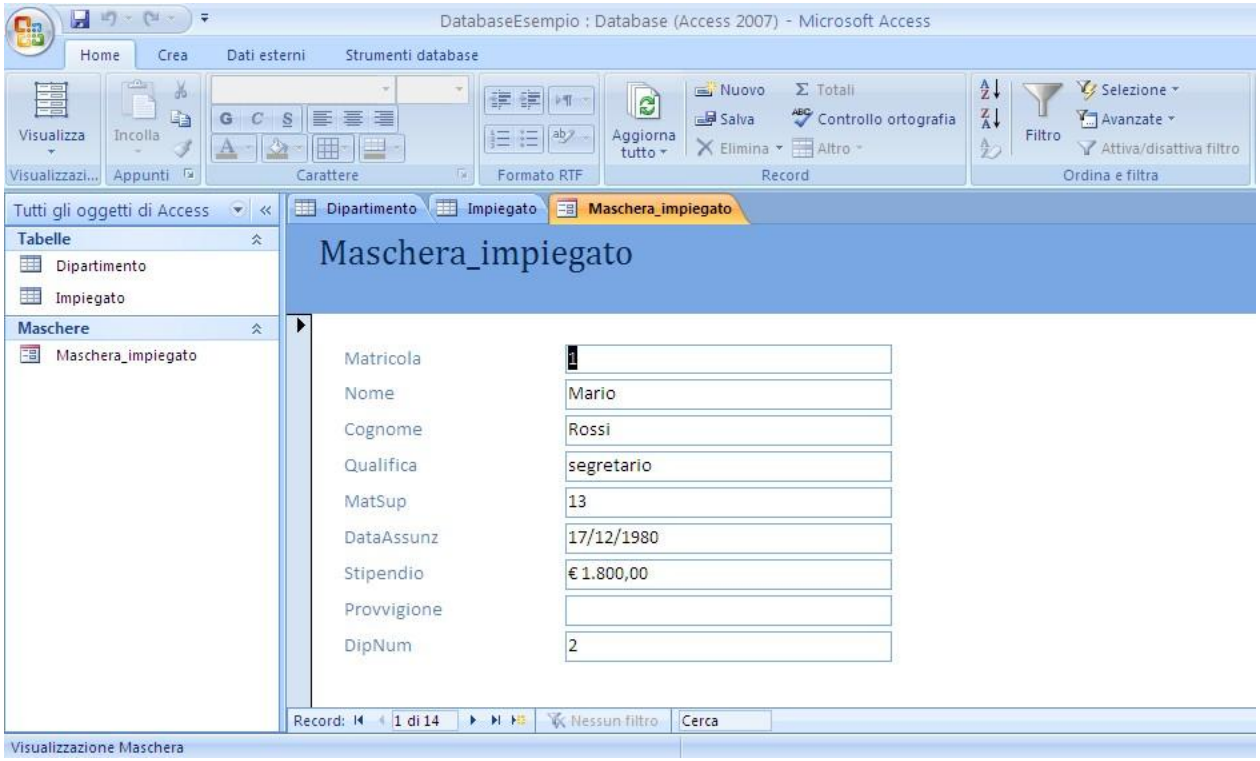


Digitiamo il nome da assegnare alla maschera appena creata, nel nostro caso assegniamo "***Maschera_impiegato"*** ed indichiamo di aprire la maschera per visualizzare o inserire informazioni (vedi figura sotto). Infine, selezionare il pulsante **Fine.**



Le maschere di Access

La figura successiva mostra la maschera generata con la creazione guidata.



Le maschere di Access

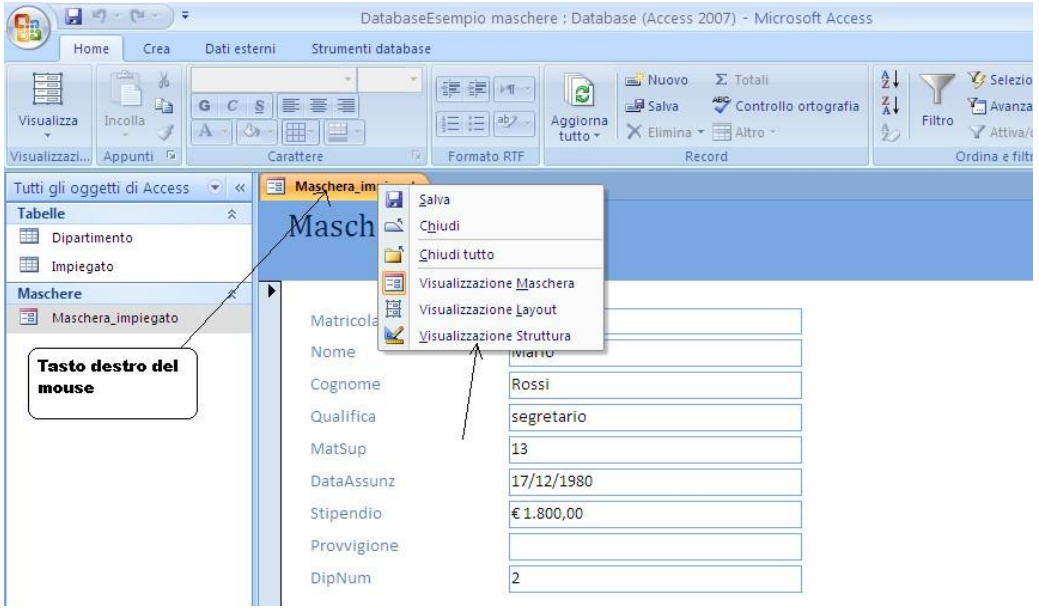
Utilizzare una maschera

Quando apriamo una maschera, sullo schermo viene visualizzato il primo record della tabella o della query associata. Per visualizzare un record specifico, si possono utilizzare i pulsanti di spostamento, posti in basso a sinistra, per passare al record successivo, precedente, primo e ultimo. Oppure si può digitare il numero del record (se conosciuto) nella casella Record specifico.

Le maschere di Access

Modificare la maschera

Le maschere possono essere in ogni momento modificate, ma per eseguire tale operazione bisogna lavorare sulla loro struttura. Quindi posizionare il mouse sull'etichetta della maschera, così come indicato dalla freccia di sinistra della figura sotto, cliccare il tasto destro del mouse, a questo punto verrà visualizzato un menu a discesa dal quale si potrà selezionare **Visualizza Struttura** che consentirà di apportare modifiche alla nostra maschera.



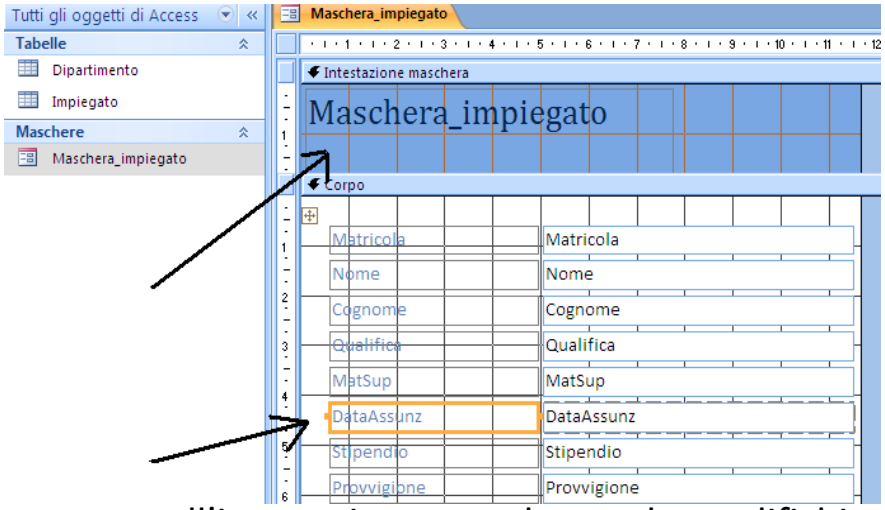
La procedura sopra descritta consente di passare dalla visualizzazione dei dati (**Visualizza Maschere**) alla visualizzazione della struttura, e viceversa, in qualsiasi momento.

La struttura si presenta come un insieme di oggetti disposti ordinatamente su una griglia, che nella visualizzazione dati non sarà comunque visibile.

Si possono selezionare gli oggetti della struttura individualmente, o a gruppi, e su di essi è possibile applicare le operazioni di formattazione.

Le maschere di Access

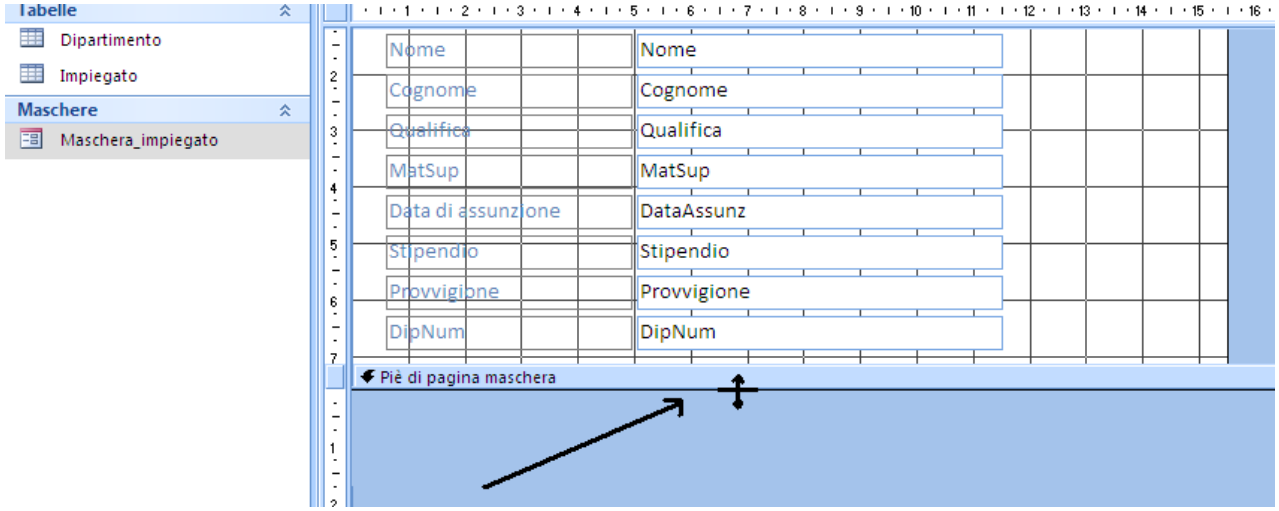
Nell'esempio successivo, modifichiamo sia l'intestazione della maschera che l'etichetta "DataAssunz"



Selezioniamo l'etichetta presente nell'intestazione maschera e la modifichiamo sia nel colore che nel testo; quindi al posto di "Maschera_impiegato" scriviamo "Menu Impiegato" e coloriamo il testo di giallo, infine mettiamo il testo al centro. Successivamente, selezioniamo l'etichetta "DataAssunz" e modifichiamo il testo con "Data di assunzione" (vedi figura sotto).

Le maschere di Access

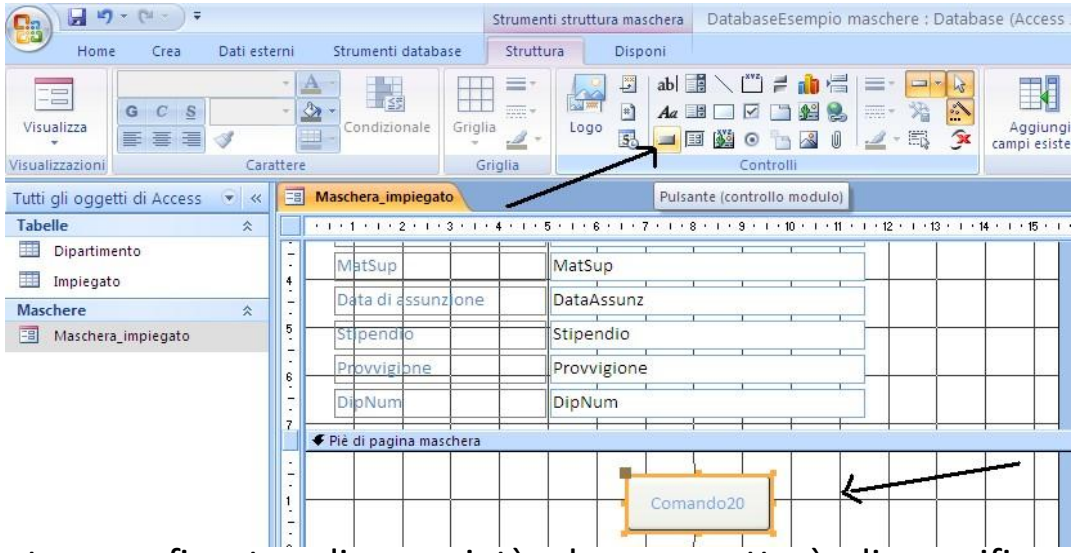
Vogliamo ora aggiungere in fondo alla maschera un pulsante che permetta la stampa di tutti i record. Quindi, posizioniamoci con il mouse sul bordo della maschera e, tenendo premuto il tasto sinistro del mouse, allunghiamo l'area dedicata al piè di pagina, così come indicato nella figura sotto.



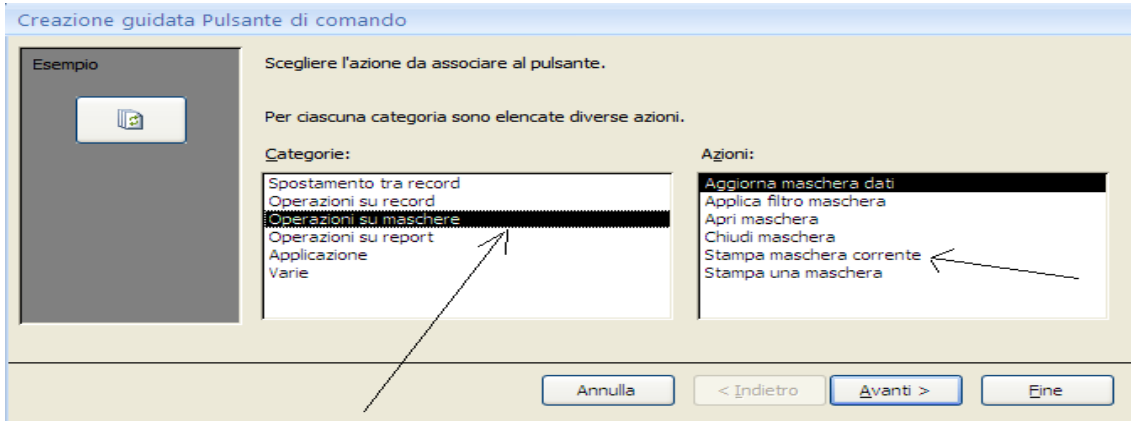
Notiamo che Access consente di aggiungere controlli come pulsanti, caselle di testo, etichette ed altri oggetti, direttamente dalla scheda **controlli**: tale scheda è presente nel menu **struttura** e viene attivata automaticamente quando ci troviamo nella modalità di **Visualizzazione struttura**.

Le maschere di Access

Quindi, una volta allargata l'area destinata al piè di pagine, inseriamo dalla scheda controlli l'oggetto **pulsante (controllo modulo),** come riportato nella prossima figura**.**

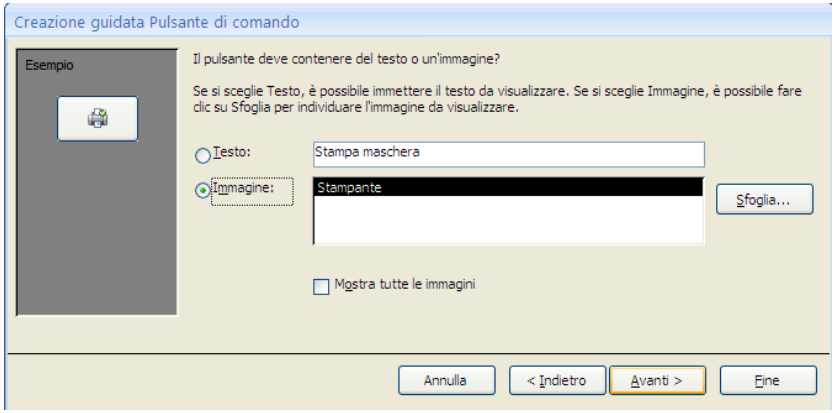


Immediatamente verrà visualizzata una finestra di proprietà che permetterà di specificare l'azione da associare al pulsante creato, quindi nel riquadro **categorie** scegliamo **Operazioni su maschere** e nel riquadro **Azioni** selezioniamo **Stampa maschera corrente**

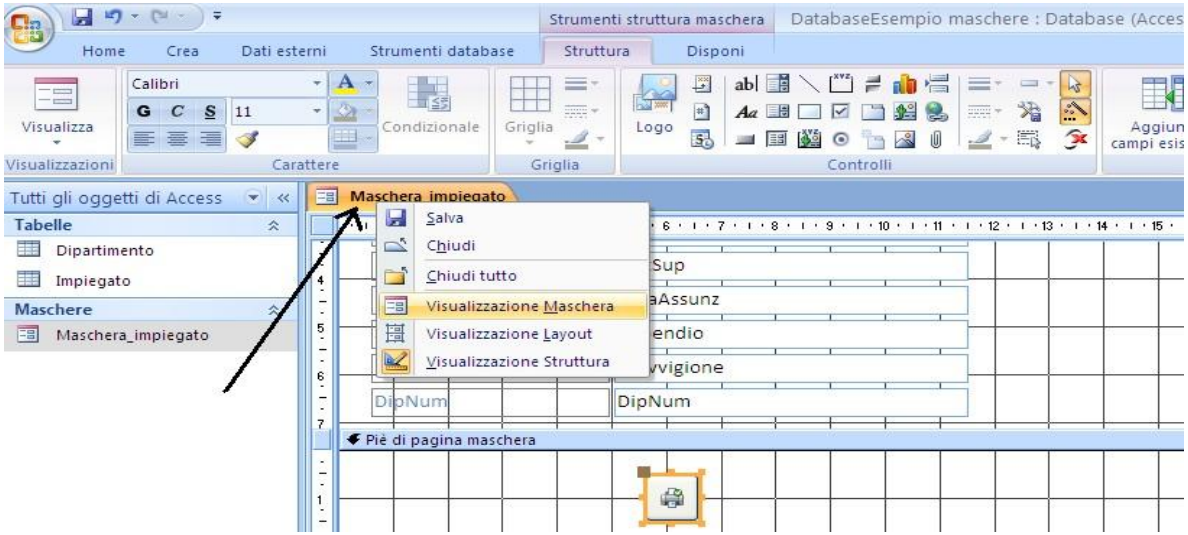


Le maschere di Access

Selezionare la casella **Immagine,** in quanto vogliamo inserire nel pulsante un'icona che rappresenti una stampante, premiamo **Avanti** e successivamente **Fine** per terminare la procedura**.**

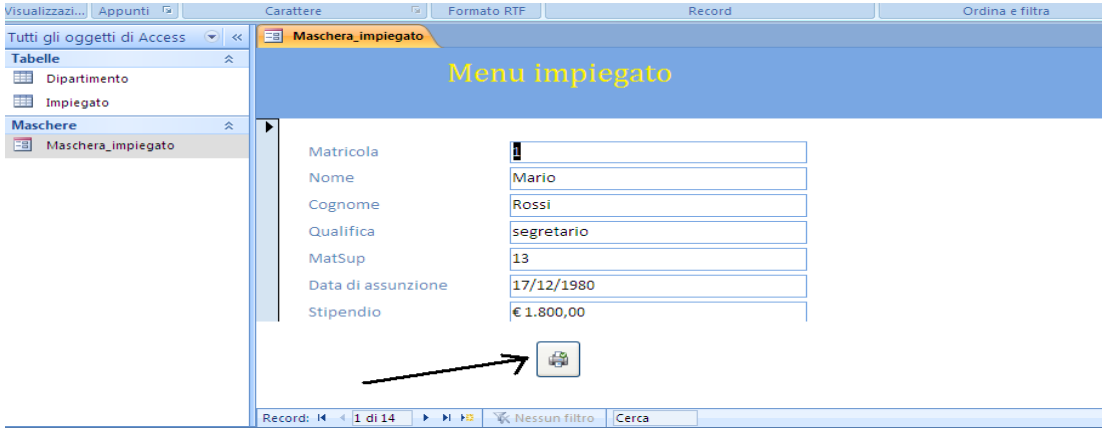


Per verificare il corretto funzionamento della maschera, posizionarsi con il mouse sull'etichetta Maschera impiegato e selezioniamo con il tasto destro la voce **Visualizzazione Maschere** (figura sotto).

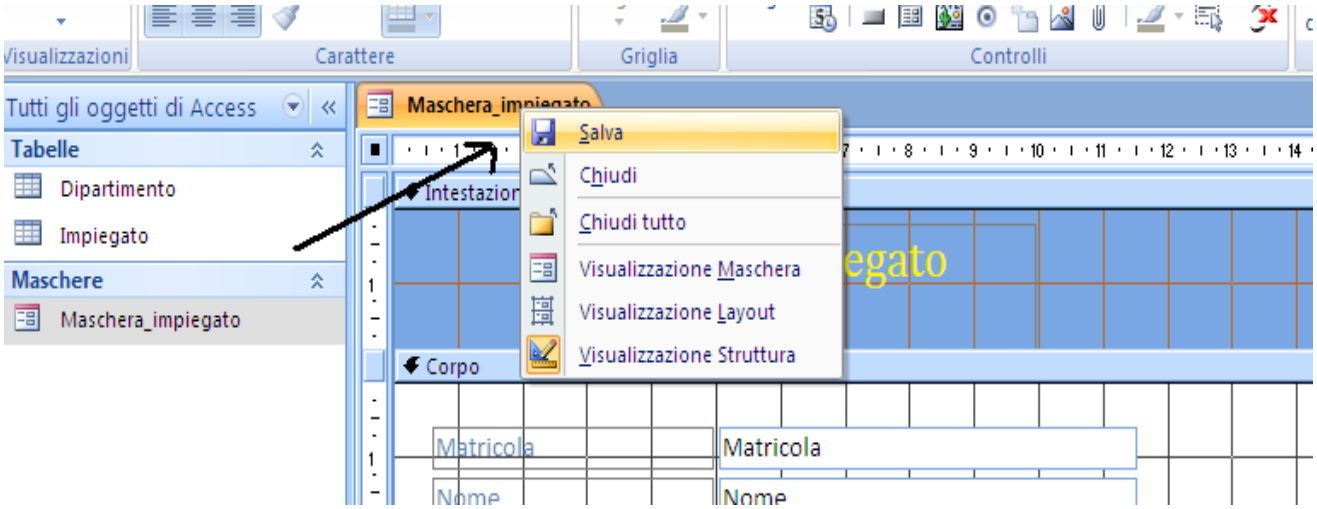


Le maschere di Access

Proviamo il pulsante di stampa inserito nella nostra maschera.



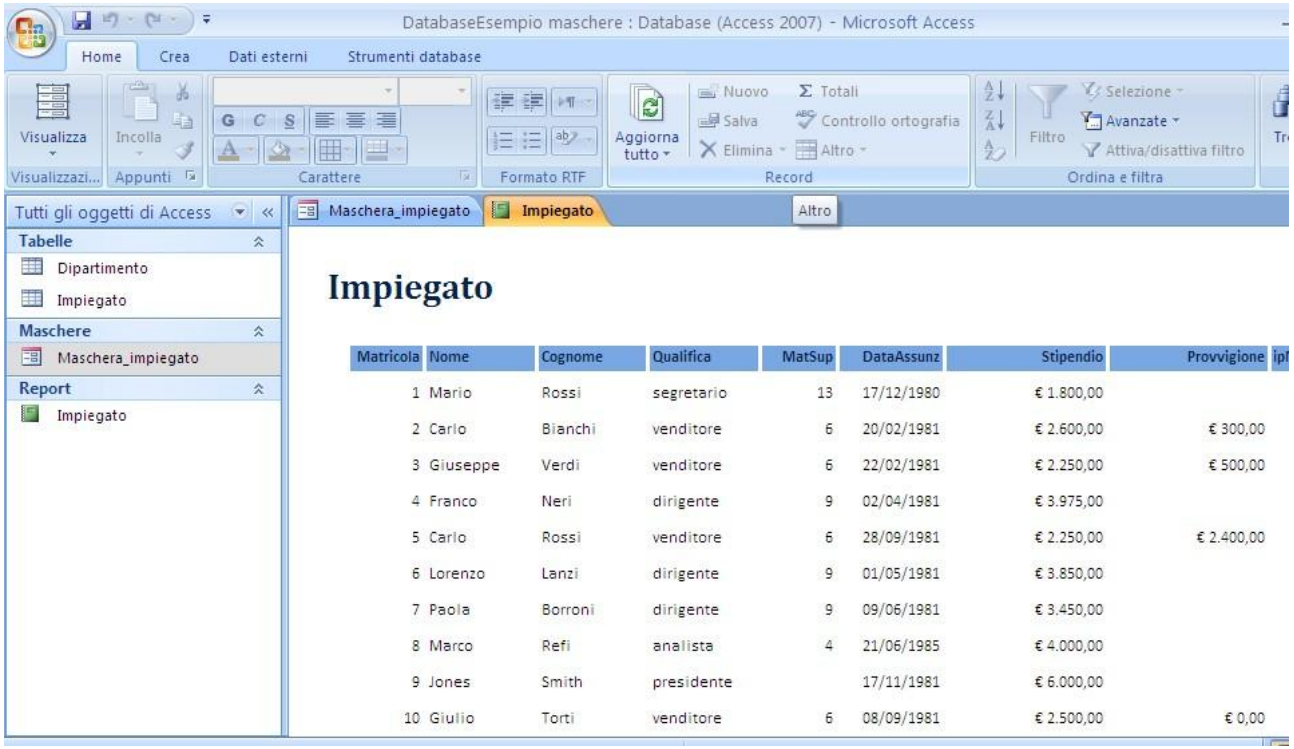
Quando si modifica la struttura della maschera è necessario salvare le modifiche apportate. Per salvare una maschera posizionarsi sull'etichetta **Maschera_impiegato** e selezionare **Salva,** così come indicato nella figura successiva.



Le maschere di Access

Un rendiconto , o report, permette di presentare i dati di un database in modo efficace e secondo un formato di stampa voluto. Nella figura successiva è visualizzata l'anteprima di stampa di un report.

I report riassumono informazioni contenute in una o più tabelle, sono strumenti molto versatili e completi per l'estrazione di informazioni del database. I dati inseriti in un report possono provenire da una tabella o da una query. A tali dati ne possono essere aggiunti ulteriori generati direttamente mediante la struttura stessa del report.



Impiegato

Matricola	Nome	Cognome	Qualifica	MatSup	DataAssunz	Stipendio	Provvigione
1	Mario	Rossi	segretario	13	17/12/1980	€ 1.800,00	
2	Carlo	Bianchi	venditore	6	20/02/1981	€ 2.600,00	€ 300,00
3	Giuseppe	Verdi	venditore	6	22/02/1981	€ 2.250,00	€ 500,00
4	Franco	Neri	dirigente	9	02/04/1981	€ 3.975,00	
5	Carlo	Rossi	venditore	6	28/09/1981	€ 2.250,00	€ 2.400,00
6	Lorenzo	Lanzi	dirigente	9	01/05/1981	€ 3.850,00	
7	Paola	Borroni	dirigente	9	09/06/1981	€ 3.450,00	
8	Marco	Refi	analista	4	21/06/1985	€ 4.000,00	
9	Jones	Smith	presidente		17/11/1981	€ 6.000,00	
10	Giulio	Torti	venditore	6	08/09/1981	€ 2.500,00	€ 0,00

Un report può essere creato definendo direttamente la struttura, o più semplicemente ricorrendo all'autocomposizione. Anche in questo caso la struttura generata con l'autocomposizione può essere facilmente modificata per adattarla alle effettive esigenze.

Le maschere di Access

Per creare un semplice report di Access puoi utilizzare una delle procedure guidate, le quali organizzano rapidamente i dati della tabella o della query selezionata disponendoli secondo un formato consultabile. La creazione guidata **Report Standard:a colonne** dispone verticalmente i dati di ciascun record, mentre **Report Standard:tabulare** li dispone orizzontalmente. Puoi creare un report anche utilizzando **Creazione guidata Report,** che permette di selezionare i campi e le informazioni che intendi presentare e di scegliere tra le opzioni di formattazione disponibili.

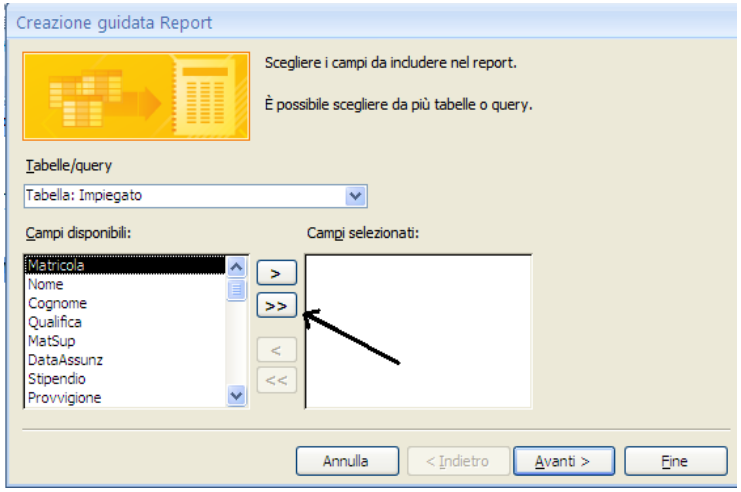
Per creare e salvare un report con una procedura guidata, procedi nel modo di seguito descritto:

1) Seleziona il menu **Crea** e nella scheda **Report**
2) seleziona **Creazione guidata Report** e apparirà la figura sotto.

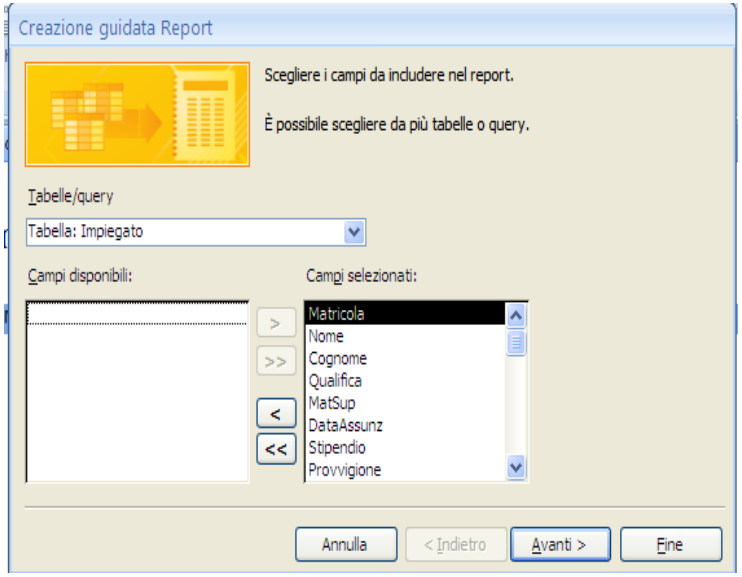
La procedura predispone una finestra di dialogo, così come indicato nella figura della pagina successiva, dalla quale sarà possibile selezionare la tabella o la query su cui basare il report; quindi selezioniamo la tabella Impiegato.

Le maschere di Access

Dal riquadro **Campi disponibili**, selezionare i campi sui quali si vuole costruire il nostro report: si può scegliere se selezionare un campo alla volta, premendo il pulsante con il simbolo **>**; diversamente, se vogliamo selezionare tutti i campi della tabella, bisogna premere sul simbolo **>>.**

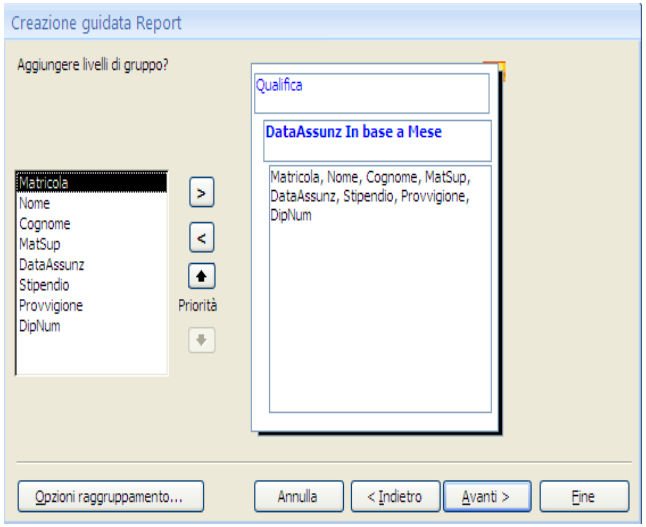


Premiamo il pulsante Avanti .



Le maschere di Access

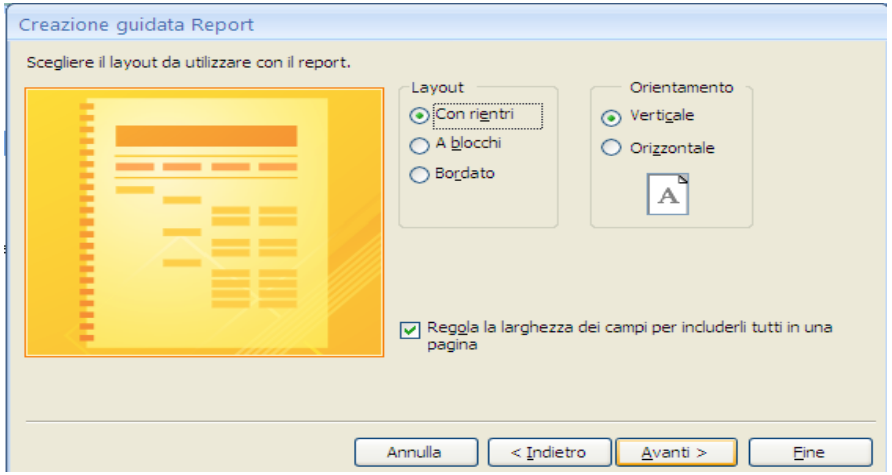
Ora bisogna specificare come raggruppare i campi, scegliendone fino a un massimo di dieci; puoi per esempio, scegliere di stampare gli impiegati divisi per Qualifica e, all'interno di questo, per data assunzione al fine di evidenziare un certo ordine nell'anzianità di servizio. Quindi per creare i gruppi, selezionare **Qualifica** e cliccare sulla **> ,** successivamente selezionare **DataAssunz** e clicca su **> ,** verrà comunque visualizzata un'anteprima nella parte destra della finestra (figura sotto).



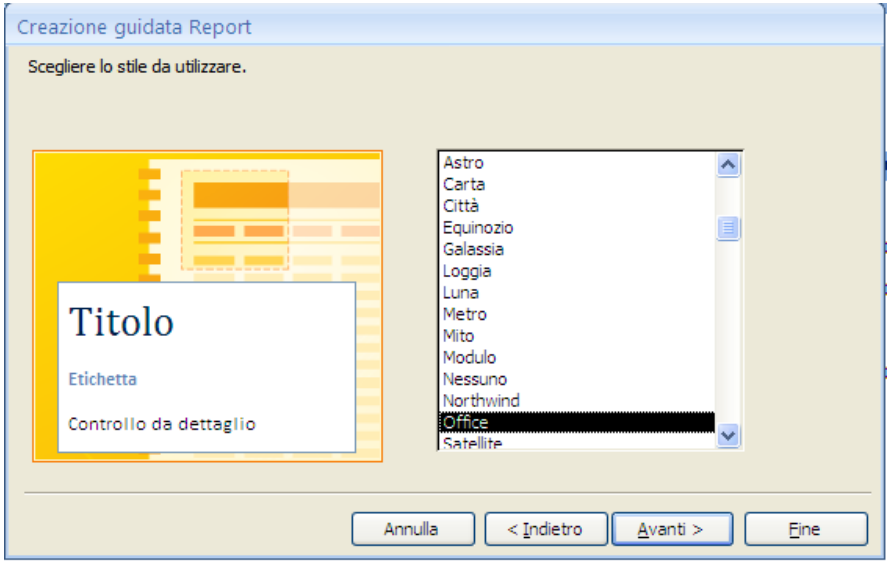
La finestra di proprietà successiva consente di ordinare i record in base ad un massimo di quattro campi, in modo crescente o decrescente. Pertanto, inseriamo il campo DataAssunz in modo crescente in quanto vogliamo visualizzare il report ordinato per data assunzione. Quindi cliccare su **Avanti**.

Le maschere di Access

Si precisa che spesso non è necessario raccogliere le informazioni in gruppi, il tutto dipende dalle esigenze dell'utente. Scegliere il layout e l'orientamento dei report e clicca su **Avanti**.

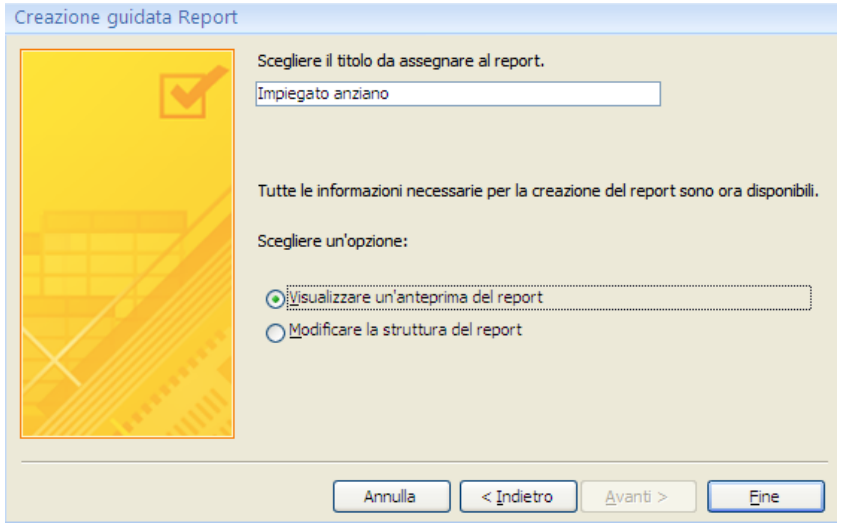


Scegliere lo stile da utilizzare, quindi cliccare su **Avanti**.

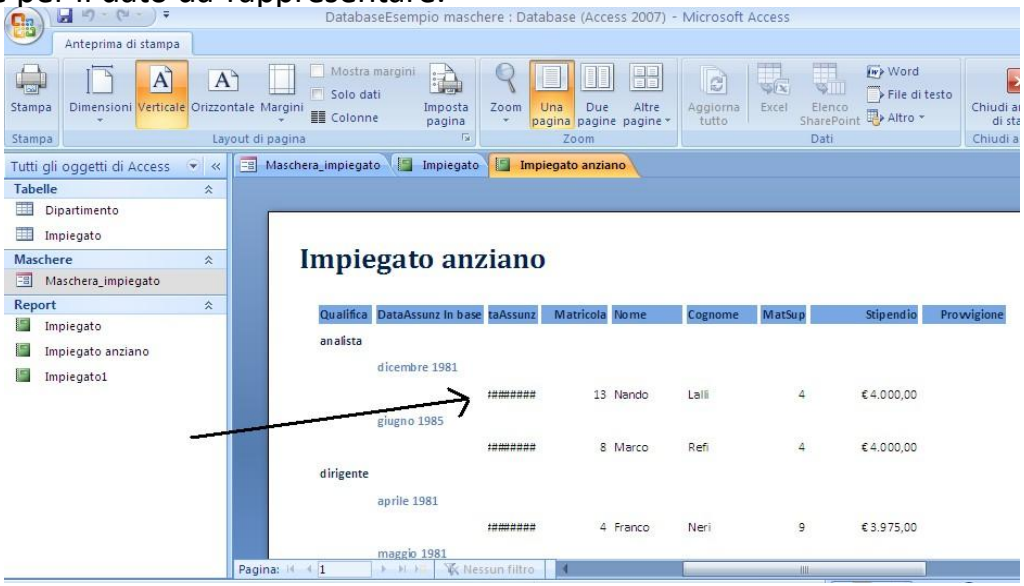


Le maschere di Access

Nell'ultima finestra di dialogo assegnare un nome (nell'esempio digitare "**Impiegato anziano**") al report ed indicare se vuoi visualizzare l'anteprima o la struttura. Infine, clicca su **Fine.**



Il report risultante è riportato nella figura sotto. Si può notare che il campo DataAssunz è troppo corto per contenere il dato , pertanto dobbiamo tornare in **Visualizzazione Struttura** e andiamo ad allargare l'oggetto **DataAssunz** fino ad ottenere una larghezza sufficiente per il dato da rappresentare.

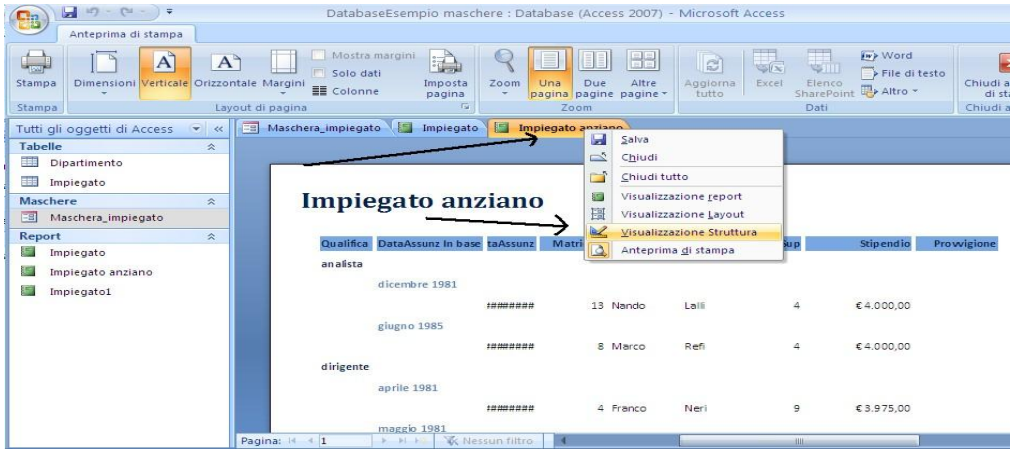


Le maschere di Access

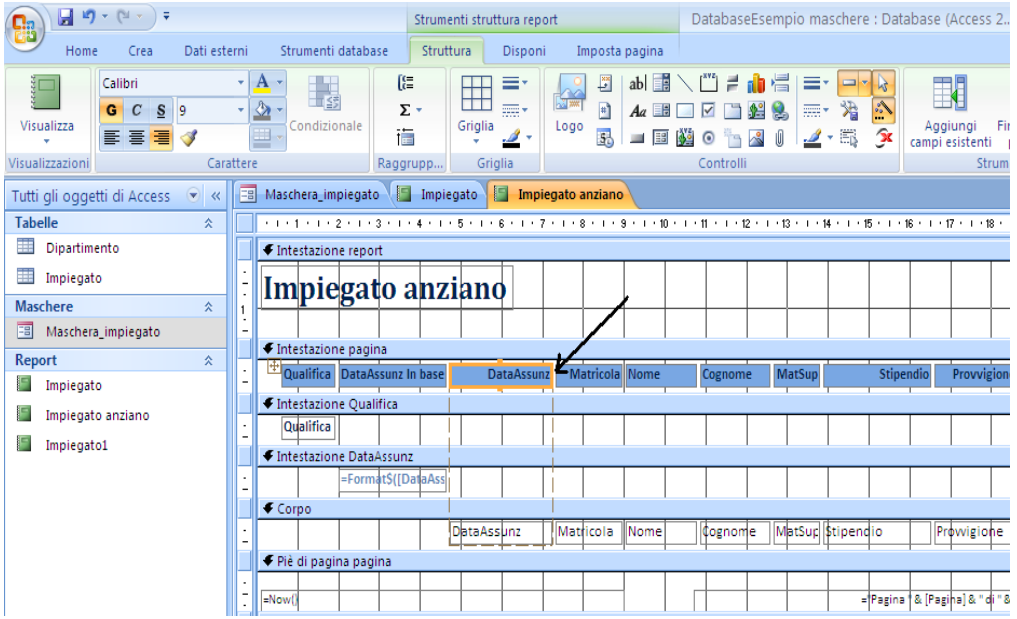
Agli oggetti presenti nel report è possibile applicare, con le stesse modalità, tutte le variazioni o modifiche viste in precedenza per le maschere.

Procediamo quindi con la modifica del campo DataAssunz.

Selezionare l'etichetta del report **Impiegato anziano,** quindi selezioniamo con il tasto destro del mouse **Visualizza Struttura.**



A questo punto, selezioniamo solo la casella **DataAssunz** e allunghiamo il bordo destro del rettangolo, così come indicato nella figura sotto.



Le maschere di Access

Quindi ritorniamo sull'etichetta **Visualizza report** e notiamo che ora la casella contiene la data di assunzione visualizzata correttamente. (figura successiva)

Come già anticipato, il report **Impiegato anziano** è strutturato in ordine alfabetico crescente sul campo **Qualifica** ed i raggruppamenti interni sono ordinati in base alla data di assunzione, così come descritto nella fase di progettazione.