D

Microsoft Access

Access, prodotto dalla Microsoft, è il più diffuso sistema di gestione di basi di dati per l'ambiente Microsoft Windows. La descrizione si baserà sulla Versione 2.0.

Access può essere utilizzato in due modalità:

- Come gestore di basi di dati autonomo su personal computer;
- Come interfaccia verso altri sistemi.

Come gestore di basi di dati autonomo risente dei limiti dell'architettura dei personal computer: offre un supporto transazionale limitato, con meccanismi di sicurezza, protezione dei dati e gestione della concorrenza piuttosto semplici ed incompleti. D'altra parte ha un costo assa ridotto, che giustifica queste limitazioni. L'interfaccia del sistema sfrutta appieno le potenzialità dell'ambiente grafico e offre un ambiente all'avanguardia, sia per l'utente applicativo che per il progettista della base di dati.

Quando è usato come client di database server relazionali, Access mette a disposizione le proprie sofisticate funzionalità di dialogo per l'interazione con questi sistemi. Access in questo contesto può essere visto come uno strumento che permette di evitare di scrivere codice SQL, in quanto acquisisce schemi e semplici interrogazioni tramite una rappresentazione grafica facilmente comprensibile; questi input vengono tradotti in opportuni comandi SQL in modo trasparente. Il protocollo ODBC, descritto nel capitolo 10, viene usato per la comunicazione tra Access e il database server.

La descrizione di Access si concentrerà sulle funzioni di gestore di basi di dati, ponendo particolarmente l'accento sulle funzionalità di definizione di schemi e interrogazioni.

D.1 Caratteristiche del sistema

Il sistema viene attivato nel modo tradizionale in cui partono le applicazioni Windows, ovvero selezionando l'icona del programma; l'icona è rappresentata in figura D.1.

All'avvio, il programma presenta una finestra vuota con una coppia di menu a tendina. Il menu File contiene i comandi New Database e Open Database.

Figura D.1 L'icona del programma

Con questi comandi si crea un database vuoto o se ne apre uno preesistente. Ad ogni database corrisponde un file con estensione standard MDB; per aprire un database preesistente bisogna selezionare il corrispondente file. Il database appena creato o aperto è rappresentato da una piccola finestra, che compare all'interno della finestra principale dell'applicazione. La figura D.2 illustra questa situazione. La finestra principale contiene un insieme di menu a tendina (File, Edit, ecc.) e immediatamente sotto una riga che contiene un insieme di bottoni, premendo i quali vengono eseguiti particolari comandi. La riga di bottoni viene detta *barra degli strumenti* o *toolbar*. I menu e la barra degli strumenti variano al variare della finestra in primo piano all'interno di Access. La figura D.2 mostra il contenuto della barra degli strumenti quando è selezionata la finestra che rappresenta la base di dati.

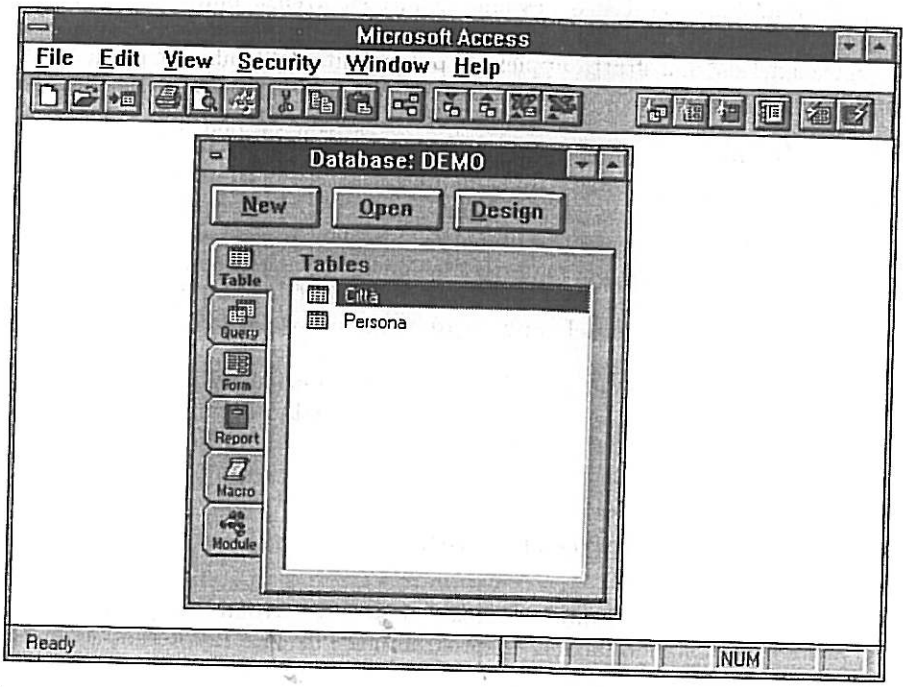

Figura D.2 La finestra contenente gli elementi del database

La finestra della base di dati contiene sul lato sinistro un elenco delle famiglie dei componenti del sistema: *tabelle*, *query*, *form*, *report*, *macro* e *moduli*. In ogni momento la finestra mostra una lista di elementi di una di queste famiglie. Per passare da una famiglia all'altra, si preme sul bottone corrispondente. Quando una famiglia è stata selezionata, nella finestra compaiono tutti i nomi degli esemplari della famiglia presenti nel database (se vi sono più esemplari di quanti la finestra ne possa contenere, viene automaticamente aggiunta una barra di scorrimento con la quale è possibile navigare lungo la lista). In figura si vede l'insieme di esemplari della famiglia *table* presenti nella base di dati di riferimento, cioè un insieme di nomi di tabelle che descrivono informazioni di tipo anagrafico.

A questo punto, per ogni famiglia, è possibile premere uno dei tre bottoni che compaiono sulla parte alta della finestra.

- Con il bottone NEW si crea un nuovo elemento della famiglia;
- Con il bottone OPEN si visualizza il contenuto di un elemento selezionato;
- Con il bottone DESIGN si riapre la fase di progetto su un elemento selezionato.

I comandi NEW e DESIGN vanno quindi ad accedere alle informazioni di progetto del componente (al suo schema), mentre OPEN accede al suo contenuto (alla sua istanza).

Descriviamo ora la definizione di tabelle e interrogazioni, mentre tratteremo sommariamente le funzionalità offerte per la gestione di form, report, macro e moduli, rimandando il lettore interessato ai manuali che vengono forniti assieme al programma e alla guida in linea, cui si accede tramite il menu Help.

D.2 La definizione e popolazione iniziale delle tabelle

Per definire lo schema di una nuova tabella, bisogna selezionare dalla finestra del database la famiglia TABLE e premere il bottone NEW. A questo punto Access propone la scelta tra l'uso della normale interfaccia di creazione o di un "wizard", ovvero uno strumento di supporto che guida nella creazione di tabelle ponendo una serie di domande. Access propone l'uso degli wizard anche in tutti gli altri contesti di definizione dei vari componenti. L'uso di wizard è consigliabile per la creazione di componenti dalla struttura regolare oppure nel caso di utenti poco esperti; per la gestione di situazioni particolari o nel caso di utenti sofisticati conviene invece fare uso delle funzionalità complete di definizione.

Per prima cosa si definiscono gli attributi, utilizzando la finestra che compare in figura D.3.

Per ciascun attributo bisogna specificare il nome e il tipo. Vi sono poi un insieme di ulteriori informazioni definibili, che variano a seconda del tipo associato all'attributo (gli attributi vengono chiamati campi, *fields*). I tipi che possono essera associati ad un attributo ricordano i tipi dello standard SQL, con qualche arricchimento. I tipi definibili sono:

- `Text`: permette di rappresentare stringhe di caratteri (corrisponde al dominio `varchar` di SQL);

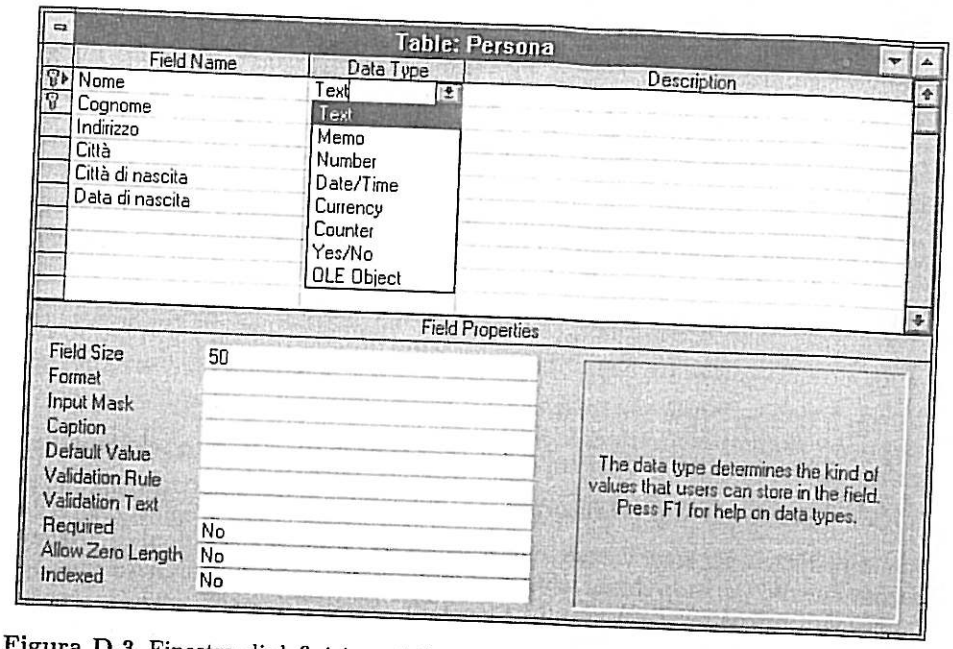

Figura D.3 Finestra di definizione dello schema delle tabelle

- `Memo`: permette di rappresentare stringhe di testo lunghe fino a 64.000 caratteri; non è un dominio esplicitamente previsto da SQL, ma può essere inteso come un caso particolare del tipo `varchar`.
- `Number`: rappresenta la famiglia di tipi numerici, interi e approssimati. Le caratteristiche aggiuntive del tipo permettono di specificare se l'attributo rappresenta valori precisi o approssimati, e il grado di precisione della rappresentazione. Corrisponde alle famiglie di domini `numeric`, `decimal`, `integer`, `smallint`, `float`, `double` e `real`.
- `Date/Time`: rappresenta istanti temporali e corrisponde ai tipi SQL `date`, `time` e `timestamp`, a seconda che si decida di rappresentare rispettivamente solo la data, solo l'ora o entrambe. La rappresentazione occupa 8 byte.
- `Currency`: serve per rappresentare valori monetari. È essenzialmente un caso particolare del tipo `Number`, caratterizzato da una rappresentazione numerica esatta su 8 byte con due cifre decimali. È rappresentato con un tipo apposito a causa della particolare rilevanza applicativa che ha la rappresentazione di valori monetari.
- `Counter`: è un particolare tipo che associa ad ogni riga della tabella un valore unico. Viene realizzato tramite un contatore incrementato ad ogni inserimento, associando un numero progressivo ad ogni riga. Usando un tipo `Counter` si ha a disposizione una chiave compatta per la tabella.
- `Yes/No`: corrisponde al tipo `bit` di SQL.

- `OLE Object`: rappresenta un generico oggetto che può essere gestito tramite OLE (Object Linking and Embedding). OLE è un protocollo che permette di specificare quale applicazione deve gestire un oggetto nell'ambiente Windows. In questo modo si possono inserire all'interno di una base di dati informazioni dei tipi più vari, come documenti di un word processor, spreadsheet, immagini, o informazioni multimediali, lasciando all'applicazione invocata tramite OLE il compito di gestirne adeguatamente il contenuto.

Per ciascun attributo si possono poi specificare un certo numero di parametri, che qualificano ulteriormente il suo dominio e la sua rappresentazione interna. I parametri compaiono nella parte inferiore della finestra di definizione dello schema.

- *Field Size*: rappresenta la dimensione del campo. Si può specificare solo per i tipi `Text` e `Number`. Per il tipo `Text` la dimensione è un valore che rappresenta la lunghezza massima della stringa di caratteri. Per il tipo `Number`, si possono specificare le seguenti dimensioni:
  - *Byte*: interi su 8 bit (valori tra 0 e 255);
  - *Integer*: interi su 16 bit (valori tra -32.768 e 32.767);
  - *Long Integer*: interi su 32 bit;
  - *Single*: rappresentazione in virgola mobile su 32 bit;
  - *Double*: rappresentazione in virgola mobile su 64 bit.
- *Format*: descrive il formato di visualizzazione dei valori degli attributi. Access utilizza dove possibile i valori specificati nell'ambiente Windows (opzioni di internazionalizzazione del pannello di controllo). Per la rappresentazione di date, numeri e valori booleani, permette di scegliere tra vari formati predefiniti (sette formati predefiniti per le date, sei per i numeri, quattro per i booleani). Si possono poi definire altri formati. È possibile inoltre distinguere la rappresentazione dei valori a seconda che siano positivi, siano negativi, siano pari a zero, o abbiano associato il valore nullo.
- *Decimal Places*: Questa informazione, definibile solo per attributi di tipo `Single` o `Double`, specifica quante cifre decimali devono essere utilizzate nella rappresentazione di default.
- *Input Mask*: Questo parametro specifica il formato che deve essere utilizzato per l'immissione dei dati. Se ad esempio un attributo registra un numero di telefono, composto da un prefisso di 3 cifre e da un numero di 7 cifre separati da un trattino, è possibile specificare una maschera di ingresso che individui le due parti e introduca immediatamente il trattino, permettendo all'utente di inserire solamente le cifre. Access mette a disposizione un wizard anche per la creazione della *Input Mask*.
- *Caption*: Rappresenta il nome che può essere dato all'attributo quando compare in una form o in un report. Può capitare di utilizzare come nome dell'attributo un nome compatto con cui sia possibile scrivere interrogazioni concise, mentre per la visualizzazione dei risultati si preferisca usare un nome che rappresenti meglio il significato dell'attributo.

- *Default Value*: Con questo parametro si specifica il valore di default per l'attributo. Corrisponde esattamente all'opzione `default` di SQL. Tutte le volte che si inserisce una nuova tupla, il valore di default comparirà come valore per l'attributo. Si può utilizzare come valore di default anche il risultato di un'espressione, come ad esempio `=Date()` che va ad assegnare a un campo di tipo `datetime` il giorno corrente.

- *Validation Rule*: Access verifica automaticamente che ogni valore inserito rispetti il tipo dell'attributo. Oltre a questo controllo, Access permette di specificare un vincolo generico per ogni attributo (in modo analogo alla clausola `check` di SQL). Questo vincolo viene espresso utilizzando la sintassi che viene utilizzata per la specifica delle condizioni in QBE, che vedremo nei prossimi paragrafi.

- *Validation Text*: Specifica il messaggio che deve essere visualizzato quando viene inserito un valore che non rispetta la regola di validazione.

- *Required*: Specifica se deve essere immesso un valore per l'attributo, ed è una proprietà di tipo booleano. Corrisponde al vincolo `not null` di SQL.

- *Allow Zero Length*: è una proprietà che vale solo per gli attributi di tipo `Text` e `Memo`. Specifica se devono essere ammesse stringhe di lunghezza nulla, o se una stringa di lunghezza nulla deve essere considerata come un valore nullo. A seconda dei contesti applicativi, può essere utile gestire in modo diverso le stringhe vuote dal valore nullo. Si tenga presente che il valore nullo viene trattato in modo particolare da SQL, per cui un confronto di disuguaglianza su stringhe viene soddisfatto da una stringa di lunghezza zero, ma non da una stringa nulla.

- *Indexed*: Mediante questa proprietà si specifica se deve essere costruito o meno un indice sull'attributo. Le opzioni possibili sono NO, YES (DUPLICATES OK) e YES (NO DUPLICATES). La terza opzione definisce sull'attributo un indice di tipo *unique*. Questo è anche il modo con cui si rappresentano i vincoli di tipo `unique`. Non si possono definire indici su attributi `Memo`, `Counter`, `Yes/No` e `OLE`. Con questa modalità si permette solo la definizione di indici semplici; per la definizione di indici più complicati bisogna operare al livello di tabella.

Dopo aver definito i vari attributi, si completa la sessione di definizione indicando quali attributi devono essere considerati la chiave primaria della tabella. L'indicazione degli attributi di chiave avviene selezionando gli attributi e premendo il bottone che raffigura la chiave nella barra degli strumenti (o selezionando l'opzione Set Primary Key nel menu Edit). Gli attributi che costituiscono la chiave vengono rappresentati con un simbolo di chiave nella colonna che precede il nome. Automaticamente Access definirà un indice di tipo `unique` sugli attributi che costituiscono la chiave.

Si possono poi definire ulteriori proprietà a livello di tabella (cui si accede tramite l'opzione Table Properties del menu View). Queste sono:

- *Description*: Una descrizione testuale del contenuto della tabella;

- *Validation Rule*: Un vincolo che deve essere soddisfatto da ogni tupla della tabella. A livello di tabella si definiscono vincoli che coinvolgono diversi attributi. Pure in questo caso, la sintassi è quella usata per l'espressione delle condizioni sulle interrogazioni. Il controllo viene effettuato al termine dell'inserimento di ogni tupla.
- *Validation Text*: Rappresenta il messaggio che viene visualizzato quando il sistema rileva una violazione del vincolo.

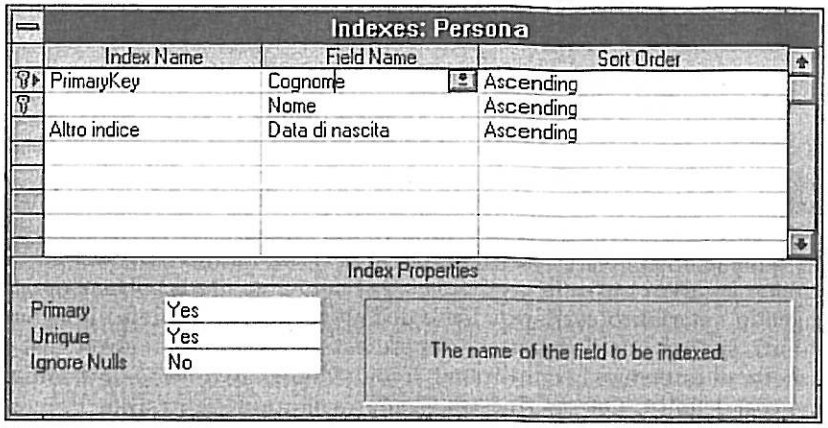

Figura D.4 La finestra di descrizione degli indici

Per specificare indici su più attributi si deve aprire la finestra di definizione degli indici, premendo il bottone INDEXES sulla barra degli strumenti, o selezionando la voce Indexes dal menu View. La finestra contiene una tabella (mostrata in figura D.4) con colonne Index Name, Field Name e Sort Order. Per definire un indice su più attributi, si inserisce in una riga il nome dell'indice, il nome del primo attributo e la direzione di ordinamento. Nella riga successiva si lascia vuoto il nome dell'indice e si introduce il nome del secondo attributo e la corrispondente direzione di ordinamento, proseguendo allo stesso modo per tutti gli attributi che caratterizzano l'indice.

Prima di terminare la sessione di definizione bisogna salvare il risultato (selezionando l'opzione Save del menu File); Access a questo punto chiede quale nome deve essere associato alla tabella. Il nome può anche contenere degli spazi al suo interno.

D.2.1 Specifica dei cammini di join

Access permette di predefinire i cammini di join che devono essere usati per formulare query su più tabelle. Per ogni cammino di join, è inoltre possibile specificare se vi è associato un vincolo di integrità referenziale. Tutto questo avviene senza che venga immesso del testo, operando esclusivamente in modo grafico.

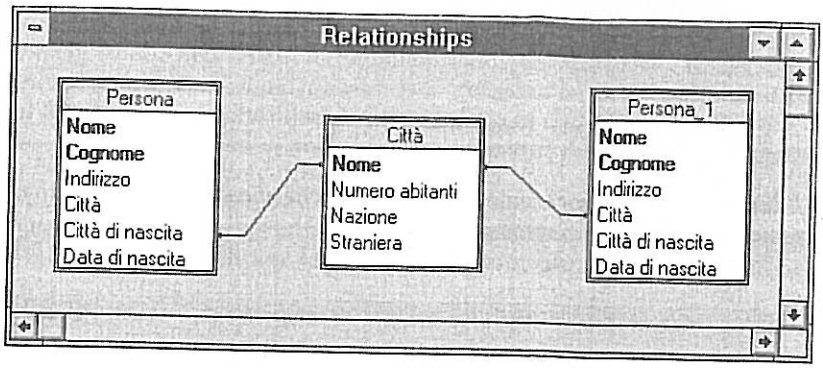

Figura D.5 La finestra di definizione dei cammini di join

Si inizia selezionando l'opzione Relationships dalla voce Edit della barra. Si apre a questo punto una finestra che contiene gli schemi delle tabelle create, rappresentata in figura D.5. È possibile definire i cammini di join tra gli schemi selezionando un attributo di uno schema e, tenendo premuto il bottone del mouse, raggiungendo l'attributo corrispondente dell'altra tabella. Definito il legame tra gli attributi, si apre una finestra che rappresenta gli attributi coinvolti nel join e che permette di estendere la condizione di join ad altri attributi o di modificare la relazione così definita. Questa finestra è rappresentata in figura D.6. Premendo il bottone JOIN TYPE di questa finestra, si ha la possibilità di scegliere quale tipo di join (*inner*, *outer left* o *outer right*) deve essere utilizzato nel legame tra le due tabelle. In questo modo tutte le volte che si definisce una query che accede alle due tabelle, verranno assegnate di default le condizioni di join specificate.

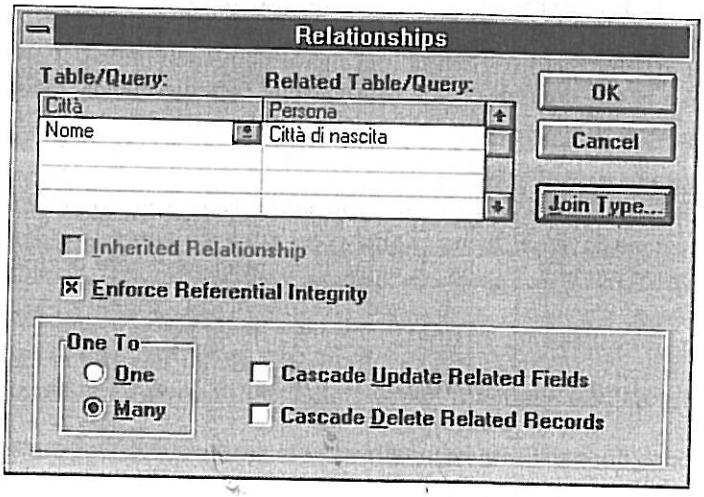

Figura D.6 La finestra di definizione delle proprietà dei join

Definito il cammino di join, è possibile specificare se al cammino deve essere associato un vincolo di integrità referenziale. Per questo bisogna dichiarare se la relazione rappresentata dal vincolo è una relazione uno a uno o uno a molti, e se si vuole che Access imponga il rispetto dell'integrità rispetto al vincolo. Nella stessa finestra, Access permette di definire una politica di reazione alle violazioni, per cui si può imporre che le modifiche o le cancellazioni siano seguite da corrispondenti modifiche e cancellazioni nelle altre tabelle. Se la politica non è specificata, ogni modifica che introduca una violazione viene semplicemente impedita. Rispetto allo standard SQL, Access permette quindi un insieme di reazioni più limitato, corrispondente alle sole scelte di `cascade delete` e `cascade update`.

Specificando i vari vincoli di integrità referenziale e applicando correttamente i criteri di progettazione descritti nel capitolo 6, si dovrebbe ottenere un grafo connesso, in cui tutte le tabelle sono collegate tramite almeno un arco con un'altra tabella dello schema. Nel caso più semplice il grafo è aciclico; ogni coppia di tabelle viene cioè collegata tramite un solo cammino. Tuttavia, in molti casi reali il grafo ha dei cicli, ed in tal caso la costruzione automatica del testo di una interrogazione a partire dai cammini di join è più complessa e talvolta arbitraria. Access non permette che nel grafo compaia più di un cammino tra due tabelle; se si devono rappresentare più cammini di join tra due tabelle, bisogna introdurre più esemplari della stessa tabella del grafo (come avviene in figura D.5).

D.2.2 Popolamento delle tabelle

Dopo aver definito lo schema delle tabelle, si può popolare l'istanza della base di dati. Anche in questo compito Access fornisce un'interfaccia grafica molto facile da usare. Aprendo una tabella dalla finestra di partenza con la modalità di default Open, compare una rappresentazione tabellare, ovvero una griglia con colonne con intestazione pari ai nomi degli attributi, e tante righe quante ne sono presenti nella tabella, a cui si somma una riga vuota che serve per inserire nuove righe nella tabella. La figura D.7 rappresenta questa finestra.

L'inserimento avviene ponendo il cursore nell'ultima riga e fornendo un valore per ogni attributo. Se il valore inserito non rispetta il tipo o un qualsiasi vincolo definito sull'attributo, l'inserimento viene immediatamente rifiutato. Spostando il

Figura D.7 La finestra che visualizza l'istanza della tabella

cursore al di fuori della riga, si indica implicitamente che l'inserimento è terminato. A questo punto il sistema controlla che i vincoli siano rispettati, ovvero che i campi per cui è necessario fornire un valore siano stati specificati e che i valori specificati rispettino le regole di validazione definite. Per effettuare delle modifiche di attributi basta andare a selezionare il valore da modificare con il cursore ed inserire il nuovo valore. Terminata l'immissione del valore e spostato il cursore in una diversa posizione, la modifica è completata.

D.3 La definizione di query

Per la definizione di interrogazioni, Access mette a dispozione due diversi strumenti: uno strumento grafico di formulazione di interrogazioni di tipo QBE (Query By Example) e un interprete SQL. Descriviamo dapprima le caratteristiche dell'interfaccia QBE, analizzando infine l'interprete SQL.

D.3.1 Query By Example

Il nome QBE fa riferimento ad una famiglia molto vasta di linguaggi di interrogazione per basi di dati, presenti in diversi sistemi in varie forme, ma tutti basati sull'idea che un'interrogazione venga formulata descrivendo le caratteristiche che devono essere possedute dalle righe del risultato. Ciò avviene riempiendo uno schema di tabella con tutti gli attributi e le condizioni che caratterizzano una riga "esemplare" del risultato.

Per definire una nuova interrogazione si seleziona l'opzione Query della finestra di partenza del database. Premendo il bottone NEW si apre la finestra di progettazione di query. Questa finestra è divisa in due metà (si vedano le figure D.8 e D.12). La metà superiore è inizialmente vuota e viene riempita con una descrizione degli schemi delle tabelle selezionate da una lista. Le tabelle vengono rappresentate con in evidenza i cammini di join predefiniti (nella metà superiore della finestra compare in effetti la porzione del diagramma di relazioni tra le tabelle di pertinenza per l'interrogazione che si deve definire). Nella metà inferiore della finestra compare una tabella inizialmente vuota, con un insieme di colonne senza nome e con righe etichettate FIELD, SORT, SHOW e CRITERIA.

- Le celle sulla riga etichettata con FIELD contengono i nomi degli attributi delle tabelle dello schema.
- Le celle della riga SORT possono essere vuote o contenere una tra le opzioni Ascending e Descending. Quando la riga non è vuota viene imposto un ordinamento delle tuple del risultato, secondo i valori dell'attributo associato alla colonna in cui compare l'opzione. Se vi sono più colonne con la cella SORT attivata, si applica per primo l'ordinamento che compare più a sinistra nelle colonne; a pari valori dell'attributo, si applicheranno man mano le successive condizioni di ordinamento, procedendo da sinistra a destra.
- Le celle della riga SHOW contengono un quadrato che può o meno contenere una croce. Se il quadrato contiene la croce, l'attributo che compare nella co-

lonna dovrà far parte del risultato della interrogazione. Lo stato del quadrato commuta ponendovi il cursore e quindi premendo il bottone del mouse.

- Le celle della riga CRITERIA contengono le condizioni che devono essere soddisfatte dalle tuple risultato della interrogazione. Le condizioni possono essere semplici confronti tra il valore dell'attributo che compare in cima alla colonna e una costante; in questo caso basta inserire il valore della costante nella cella. La condizione può anche essere più complicata e includere riferimenti ad altri attributi.

Dopo che la query è stata formulata, per eseguirla bisogna premere il bottone della barra degli strumenti che contiene il punto esclamativo; dopo l'esecuzione, la tabella risultato della query compare al posto della finestra di definizione della query. Per visualizzare i nomi degli attributi in corrispondenza alle colonne si possono utilizzare due modalità: si può scrivere direttamente il nome dell'attributo, eventualmente qualificato col nome della tabella di appartenenza, o si possono selezionare gli attributi che compaiono nella rappresentazione degli schemi della metà superiore della finestra, "trascinandoli" nelle relative colonne.

Vediamo alcuni esempi di definizione di interrogazioni. Supponiamo di avere una base di dati con una tabella PERSONA(Nome,Cognome, Indirizzo, Città, Città di nascita, Data di nascita) e una tabella CITTÀ(Nome, Numero abitanti). Per trovare i nomi, ordinati alfabeticamente, delle persone aventi cognome "Rossi" possiamo riempire lo schema nel modo descritto in figura D.8.

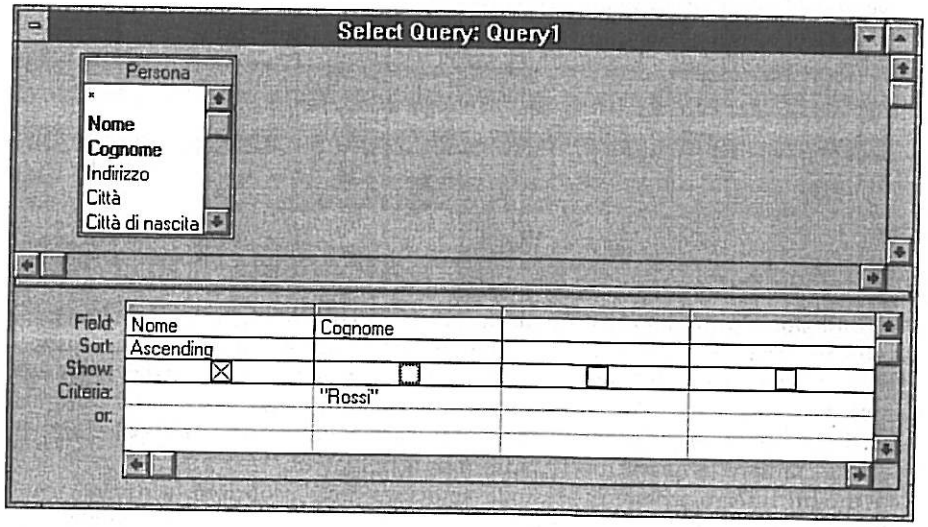

Figura D.8 Interrogazione QBE che restituisce i nomi delle persone di cognome "Rossi"

Per formulare più condizioni in congiunzione, si riempiono più campi della riga CRITERIA. Se bisogna selezionare le tuple che soddisfano più condizioni in alternativa, bisogna riempire più righe con i diversi criteri. Access etichetta automaticamente con OR le righe aggiuntive. Così, per trovare i nomi, cognomi e

indirizzi delle persone di Milano aventi cognome "Rossi" o "Bianchi", si creerà uno schema come quello in figura D.9.

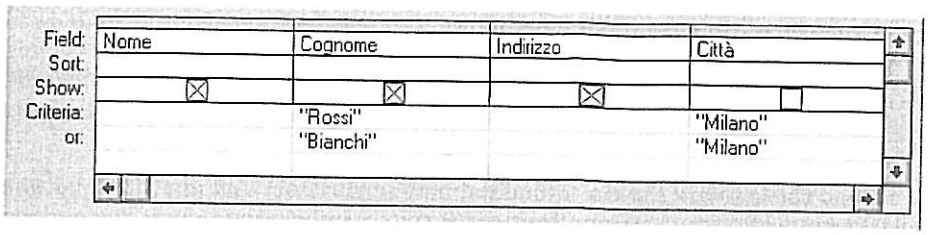

Figura D.9 Query che restituisce i milanesi chiamati "Rossi" o "Bianchi"

Tra la lista degli attributi compare anche il simbolo di asterisco che, con significato analogo a SQL, rappresenta tutti gli attributi. Per selezionare quindi tutti gli attributi delle tuple della tabella PERSONA che hanno la residenza nella città di nascita, potremo formulare la query QBE rappresentata in figura D.10.

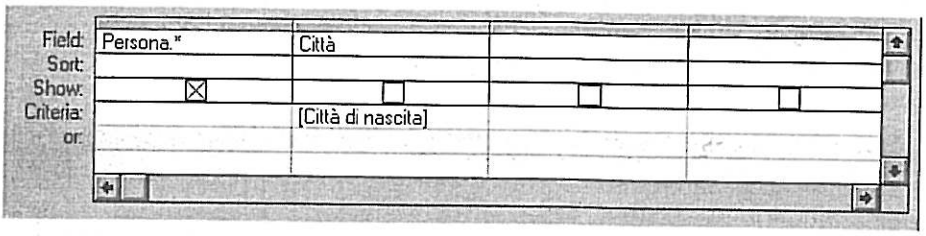

Figura D.10 Query che restituisce le persone con città di nascita e di residenza uguali

Per distinguere i valori costanti dai nomi degli attributi, bisogna racchiudere i nomi degli attributi tra parentesi quadre. Access permette poi di formulare delle condizioni utilizzando i normali operatori di confronto ($<$, $\leq$, $>$, $\geq$ e $<>$) e l'operatore `like`, per il confronto di stringhe con espressioni regolari che usano i caratteri speciali `*` e `?` (che corrispondono rispettivamente ai caratteri `%` e `_` della sintassi standard SQL). Per trovare i nomi e cognomi delle persone che sono nate prima del 31 gennaio 1965 e che hanno un cognome che inizia con la lettera 'C', si potrà formulare la query mostrata in figura D.11.

Ad ogni interrogazione è possibile associare delle proprietà, a diversi livelli. A livello di singola colonna, è possibile specificare un formato di visualizzazione diverso da quello definito nella fase di definizione dello schema della tabella. Un'altra importante proprietà è l'eliminazione di eventuali duplicati presenti nel risultato. Per specificare che i duplicati devono essere rimossi bisogna aprire le proprietà della query (tramite l'opzione PROPERTIES del menu View) e assegnare alla proprietà UNIQUE VALUES il valore *Yes*.

Per formulare delle interrogazioni che richiedono più tabelle, è opportuno far comparire nella finestra superiore le tabelle richieste (selezionandole dalla fine-

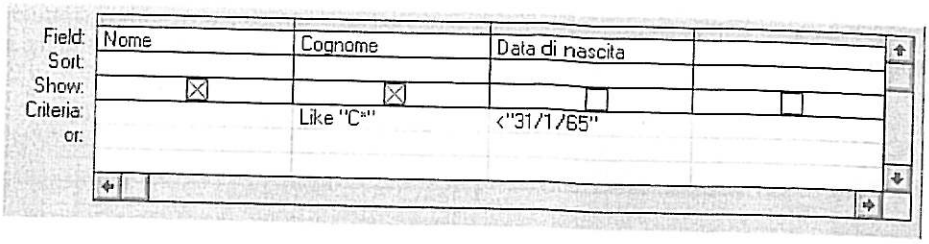

Figura D.11 Query che restituisce i nomi e cognomi delle persone con un cognome che inizia per "C" nate prima del 31/1/65

stra di dialogo che compare appena si crea una nuova interrogazione). Le tabelle compariranno legate dai cammini di join che sono stati definiti al momento della definizione dello schema. Nella tabella che rappresenta la query converrà poi aggiungere anche un'ulteriore riga che rappresenta il nome della tabella da cui viene prelevato l'attributo (la riga TABLE). Per selezionare gli attributi si può "trascinare" con il mouse l'attributo dalla metà superiore alla colonna della metà inferiore, oppure si può scrivere direttamente il nome nella cella della riga FIELD. Le condizioni di join tra le tuple delle tabelle non dovranno essere specificate se queste sono state predefinite. Quando le condizioni di join predefinite non sono quelle richieste dalla particolare query, è possibile modificare i cammini che collegano le tabelle nella rappresentazione della parte superiore della finestra.

Volendo ad esempio formulare la query che trova le persone nate in una città con meno di centomila abitanti, se è stato predefinito il legame di join tra le tabelle PERSONA e CITTÀ, si può formulare la query che appare in figura D.12.

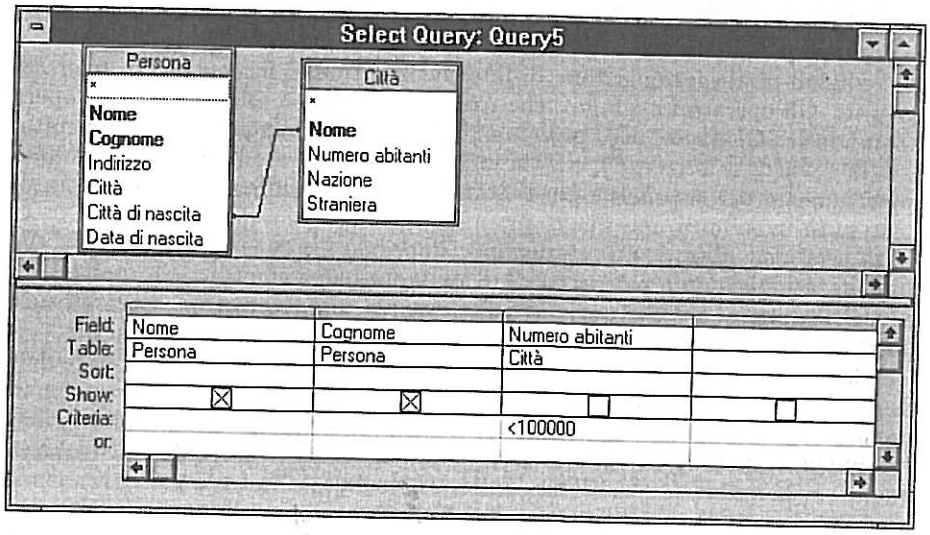

Figura D.12 Query che restituisce i nomi e cognomi delle persone nate in una città con meno di 100.000 abitanti (con join predefinito)

Qualora invece il legame di join non sia stato predefinito, si dovrà rendere esplicita la condizione nella tabella, formulando una interrogazione come quella in figura D.13.

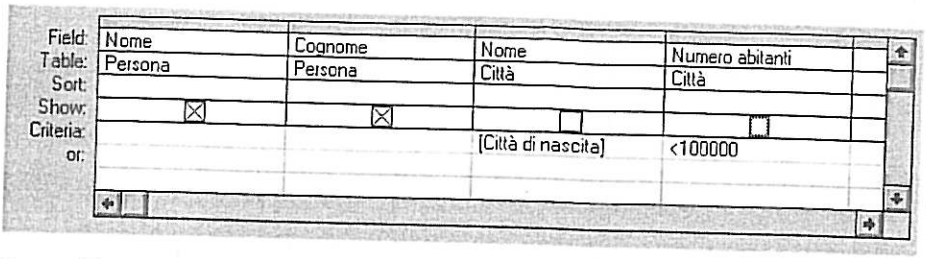

Figura D.13 Query che restituisce i nomi e cognomi delle persone nate in una città con meno di 100.000 abitanti (senza join predefinito)

Può essere necessario talvolta introdurre tra le tabelle che compaiono nella metà superiore anche tabelle di cui non vengono utilizzati attributi nella query. Un caso importante di questo tipo è quello in cui si devono estrarre informazioni tra due tabelle che non hanno un legame diretto, ma in cui il legame è realizzato tramite una tabella intermedia (come avviene quando si ha nel progetto concettuale una relazione molti a molti e si introduce nel progetto logico una tabella ponte). In questo caso, anche se l'interrogazione utilizzerà per la visualizzazone e l'applicazione di condizioni solo attributi delle due tabelle esterne, la tabella ponte dovrà comparire nella metà superiore. Il sistema infatti esegue il prodotto cartesiano delle tabelle che compaiono nella metà superiore, applicando tutte le condizioni di join che sono state definite. Il risultato di questa operazione è il punto di partenza per l'applicazione delle condizioni che compaiono nello schema della query QBE.

Vediamo ora la formulazione di interrogazioni QBE facenti uso di operatori aggregati. Gli operatori aggregati che Access mette a disposizione sono gli operatori standard SQL (*sum*, *avg*, *min*, *max*, *count*) a cui si aggiungono gli operatori *stdev* (la deviazione standard), *var* (la varianza), *first* e *last* (il valore dell'attributo rispettivamente per la prima e per l'ultima tupla). Per utilizzare questi operatori è necessario introdurre una nuova riga nello schema della query, la riga TOTAL. Questa riga viene introdotta selezionando l'opzione TOTALS dal menu View. Vediamo un semplice esempio d'uso degli operatori, definendo, come mostrato in figura D.14, una query che permette di trovare il numero di tuple presenti nella tabella PERSONA.

Il fatto che nella riga TOTAL compaia il valore *Count* fa sì che il risultato della interrogazione non siano i nomi delle tuple della tabella PERSONA, ma appunto il numero di tuple. Nella riga FIELD deve comparire uno qualsiasi degli attributi della tabella PERSONA; sarebbe stato meglio, per coerenza con SQL-2, che Access permettesse in questo contesto l'uso dell'asterisco (*), utilizzando invece il nome di un attributo per contare i diversi valori dell'attributo; purtroppo nella valutazione degli altri operatori aggregati l'asterisco crea delle complicazioni, per cui in Access si è scelta la soluzione di impedire l'uso dell'asterisco quando è abilitata la riga TOTAL.

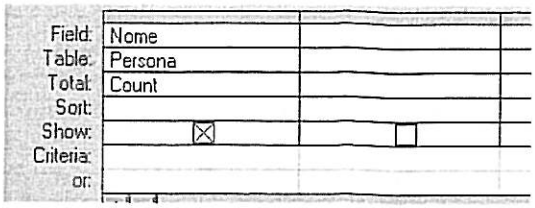

Figura D.14 Query che restituisce il numero di persone

Una interrogazione che restituisce i cognomi delle persone presenti nella base di dati, associando ad ognuno di essi il numero di persone che lo possiedono, si può formulare nel modo descritto in figura D.15.

In questo caso l'attributo Cognome viene caratterizzato dal valore *Group by* nella riga TOTAL, ovvero l'attributo viene utilizzato per raggruppare le tuple della tabella PERSONA. La seconda colonna rappresenta l'applicazione dell'operatore *Count* ad ogni singolo raggruppamento.

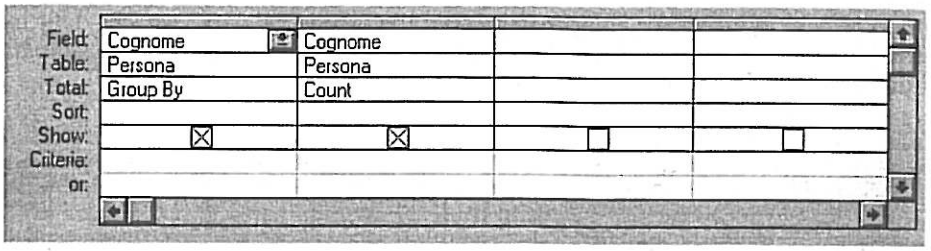

Figura D.15 Query che restituisce il numero di persone che possiede ciascun cognome

Per esprimere delle condizioni sul risultato della valutazione degli operatori aggregati (in modo analogo all'uso della clausola `having` in SQL), basta imporre le opportune condizioni nella riga CRITERIA, in modo analogo all'espressione di semplici condizioni sulle tuple. Così, per trovare i cognomi che sono posseduti da almeno due persone, si potrà scrivere l'interrogazione QBE in figura D.16, che restituisce i cognomi posseduti da più di una persona, indicando per ogni cognome il numero di volte che questo appare nella tabella PERSONA.

Se in una interrogazione con raggruppamento le tuple devono essere selezionate preliminarmente in base ai valori di attributi che non vengono utilizzati nel raggruppamento, diventa necessario distinguere le condizioni che devono essere valutate prima del raggruppamento da quelle che devono essere eseguite nella fase successiva. Questa distinzione in SQL avviene ponendo le condizioni preliminari nella clausola `where` e le condizioni successive nella clausola `having`; nel linguaggio QBE la distinzione avviene ponendo il valore *Where* nella riga TOTAL per gli attributi che servono solo per la selezione delle tuple da raggruppare. La presenza del valore *Where* è incompatibile con il flag di SHOW, in quanto nel risultato delle interrogazioni che fanno uso di operatori aggregati possono comparire solo

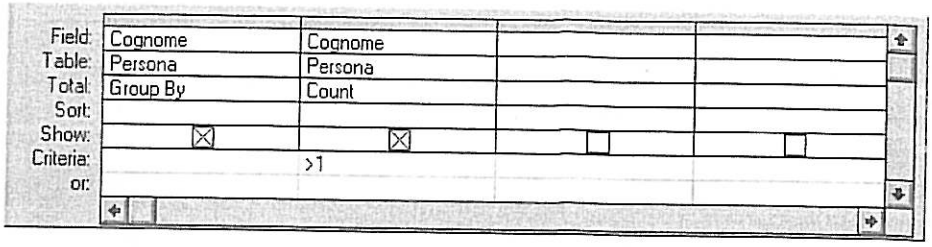

Figura D.16 Query che restituisce il numero di persone che possiedono un cognome, per i cognomi posseduti da più persone

il risultato della valutazione degli operatori aggregati e gli attributi su cui viene effettuato il raggruppamento. Per trovare gli omonimi che sono nati a Milano, si potrà formulare l'interrogazione in figura D.17.

Un modo alternativo per formulare questa interrogazione che non fa uso del termine *Where* consiste nel formulare una prima interrogazione che estrae le persone nate a Milano, e una seconda interrogazione che parte dal risultato della prima andando a selezionare gli omonimi. Ad ogni interrogazione viene infatti associato un nome e un'interrogazione può partire sia dalle tabelle del database, che dalle interrogazioni già definite. Ogni interrogazione può quindi anche essere considerata come una definizione di una vista sul database.

Figura D.17 Query che restituisce gli omonimi tra le persone nate a Milano

D.3.2 L'interprete SQL

Oltre al linguaggio di interrogazione QBE, Access fornisce un interprete SQL, che può essere usato in alternativa a QBE. Access permette di passare rapidamente dal contesto QBE al contesto SQL e viceversa, premendo gli opportuni bottoni sulla barra degli strumenti o selezionando rispettivamente le opzioni SQL e DESIGN dal menu View. Il passaggio da un ambiente all'altro trasforma l'interrogazione corrente nella corrispondente interrogazione nell'altro ambiente.

Il passaggio da QBE a SQL è sempre possibile. In effetti, tutte le volte che una query QBE viene mandata in esecuzione, essa viene prima tradotta nella corrispondente forma SQL e viene quindi eseguita dall'interprete SQL. Il passaggio

inverso non è invece sempre possibile, in quanto il linguaggio SQL è più potente di QBE, permettendo ad esempio l'espressione di query con l'operatore di unione. SQL risulta in generale più comodo per la scrittura di un'ampia famiglia di interrogazioni, come quelle che fanno uso dell'operatore `not exists`. Il linguaggio QBE è in sostanza un linguaggio molto potente e facile da usare quando si devono formulare interrogazioni che fanno uso solo di selezioni, proiezioni e join: in questo caso la possibilità di formulare le interrogazioni senza bisogno di scrivere del testo secondo una rigida sintassi, rappresenta un notevole aiuto. D'altra parte QBE non mette a disposizione un meccanismo adeguato per la rappresentazione di interrogazioni complesse, come quelle che richiedono di formulare delle interrogazioni nidificate in SQL. Di fatto, quando una interrogazione SQL che fa uso di query nidificate viene tradotta in QBE, la traduzione riporta semplicemente il testo dell'intera query nidificata nell'opportuna cella della riga CRITERIA.

Per quanto riguarda la sintassi riconosciuta dall'interprete, si tratta di un'estensione della sintassi standard SQL, con un supporto per particolari funzionalità e alcune differenze sintattiche e semantiche. Fra di esse:

- La clausola *Top* viene utilizzata per selezionare un certo numero di tuple dal risultato.
- Vengono usate le parentesi quadre per racchiudere gli identificativi di tabelle ed attributi (necessarie quando compaiono degli spazi o caratteri speciali all'interno dei nomi).
- L'operatore `join` deve essere sempre qualificato dal termine `inner` o `outer`.
- La valutazione dell'operatore `count` è diversa: se si dà come argomento un attributo, non vengono restituiti i distinti valori dell'attributo, bensì il numero di valori non nulli (come se fosse specificata implicitamente l'opzione `all`); l'opzione `distinct` non è riconosciuta.

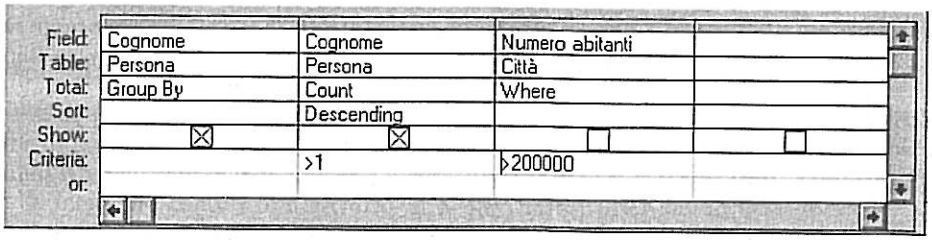

Figura D.18 Query che restituisce il numero di omonimi in città sopra i 200.000 abitanti

Ad esempio, si consideri la query QBE in figura D.18, per cui è specificata la proprietà che vengano restituiti solo i primi 10 elementi e per cui è stato predefinito un cammino di join. Ad essa corrisponde la seguente query nel dialetto SQL di Access:

```
select top 10 Cognome, count(Cognome) as NumeroOmonimi
from Persona inner join Città
                        on Persona.[Città di nascita] = Città.Nome
where [Numero abitanti] > 200000
group by Cognome
having count(Cognome) > 1
order by count(Cognome) desc
```

D.4 Form e report

Le *form*, dette in italiano *maschere*, permettono di rappresentare il contenuto del database in modo molto più chiaro e comprensibile, rispetto alla piatta rappresentazione delle righe delle tabelle.

Le form sono analoghe ai "moduli prestampati", come i moduli utilizzati per fare le richieste di certificati all'anagrafe, caratterizzati da un insieme di attributi in cui si devono inserire i dati, e un insieme di etichette che specificano quale dato deve essere inserito nel particolare attributo. Le maschere possono servire per l'inserimento dei dati, realizzando una versione elettronica del modulo prestampato, e possono anche essere utilizzate per visualizzare e modificare il contenuto della base di dati.

Gli strumenti di generazione di maschere sono sempre stati uno degli strumenti di supporto più comuni tra quelli offerti dai DBMS. In Access, al posto della semplice e tradizionale interfaccia a caratteri, viene permessa la realizzazione di sofisticate maschere grafiche. Lo strumento di definizione delle form permette di definire la posizione e il significato di ciascun componente della maschera, dando un'ampia libertà di personalizzazione nell'ambito della scelta di fonti di caratteri, colori, disegni e simboli grafici.

Per creare una nuova form, bisogna selezionare il componente FORM dalla finestra di base del database e premere il bottone NEW. Il sistema chiede a questo punto se si vuole utilizzare lo strumento di progetto di base, o se si vuole utilizzare un wizard; mediante un wizard si può creare in pochi istanti una semplice form associata ad una tabella. Access offre in questo contesto la possibilità di scegliere tra vari wizard per la creazione di form; i vari wizard differiscono in base allo schema di form che producono (per cui un wizard produce una form con tutti gli attributi della tabella in sequenza uno per riga, un altro produce una rappresentazione tabellare arricchita da spazi e colori). Se non si utilizzano i servizi di un wizard, ci si trova davanti ad una pagina bianca in cui si possono inserire man mano i componenti della form.

Sono due le modalità principali per operare sulle form. La prima è la modalità di progetto della form (in cui si entra definendo una nuova form o premendo il bottone DESIGN su una form preesistente); in questa modalità si definiscono i vari elementi della form. L'elemento di base di una form è il *control*, un oggetto cui corrisponde un'area rettangolare dello schermo e che può essere di tre tipi: *bound*, *unbound* e *calculated*. Un control *bound* è un elemento a cui viene associato un attributo di una tabella. Questo elemento rappresenterà il valore dell'attributo per la particolare tupla considerata (questo è in genere rappresentato da una sequenza di caratteri; per attributi di tipo OLE la rappresentazione viene affidata all'appli-

cazione ad esso associata). Un control *unbound* invece contiene un valore fisso, che non cambia al variare delle tuple; in questo modo si rappresentano tipicamente le etichette delle form (che anche in questo caso possono essere sequenze di caratteri, o altri oggetti di tipo generico; se si vuole ad esempio inserire un logo nella form, bisognerà associare a un control unbound la figura che rappresenta il logo). Infine, i control calcolati permettono di visualizzare i risultati di espressioni valutate a partire da parametri costanti e dai valori degli attributi delle tuple.

La seconda modalità d'uso della form consente di inserire nuove tuple, riportando direttamente i valori degli attributi negli opportuni control bound. Con questa modalità è possibile anche interrogare la base di dati; per ricercare una tupla, è sufficiente inserire il valore di uno o più attributi nella form e quindi dare il comando di ricerca (opzione SEARCH del menu View). Per modificare il valore di un attributo, si seleziona la tupla e si apporta la modifica direttamente sull'attributo. Le modifiche vengono rese persistenti spostandosi su una diversa tupla, premendo i tasti per l'avanzamento o l'arretramento di una pagina. La figura D.19 mostra una form sulle tuple di PERSONA, estese con un attributo che contiene una foto.

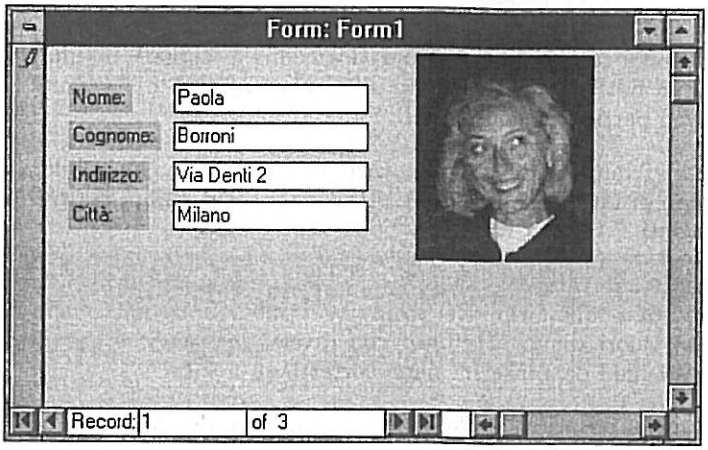

Figura D.19 Una form che permette di accedere ad alcuni attributi di PERSONA

Le form sono in genere costruite su una singola tabella. È possibile definire delle form che contengono altre form al loro interno (fino a un massimo di due livelli) specificando dei legami tra i valori visualizzati nelle form. In questo modo si possono ad esempio definire delle form per la gestione di tabelle con relazioni uno a molti, in cui la form interna rappresenta tutte le tuple della relazione di dettaglio associate all'elemento esterno (si pensi a una coppia di tabelle che descrivono gli ordini; una maschera può illustrare al livello più esterno i dati dell'ordine e in una sottomaschera la distinta degli ordinativi, appartenenti a due tabelle distinte).

Un *report* si definisce in modo analogo ad una form, ma ha l'obiettivo di fornire una descrizione del contenuto della base dati al livello globale. Per questo i report tipicamente contengono dei control calcolati che forniscono consuntivi, e

non permettono la visione di tuple particolari. Un'altra differenza con le form è che in genere i report vengono stampati, invece di essere utilizzati in modo interattivo.

D.5 La definizione di macro e la scrittura di applicazioni

Le macro costituiscono un modo per specificare un insieme di azioni che il sistema deve compiere. In questo modo è possibile automatizzare l'esecuzione di un insieme di compiti. Mediante le macro risulta possibile:

- Far interagire una form con altre form e con i report. Avendo una form che descrive i dati dei clienti, e una che descrive gli ordini, è possibile aggiungere alla prima form un bottone che attiva la seconda form visualizzando solo i dati degli ordini del cliente visualizzato. Si può anche aggiungere un bottone che permette di stampare un report che elenca la situazione contabile nei confronti del cliente, con l'ammontare di merce rispettivamente ordinata, consegnata e pagata.

- Selezionare e raggruppare automaticamente le tuple. In una form si può inserire un bottone che permette di selezionare immediatamente le tuple che rispettano particolari condizioni.

- Assegnare i valori agli attributi. Usando una macro, è possibile assegnare ad un campo di una form un valore ottenuto da altri campi o altre tabelle della base di dati.

- Garantire l'accuratezza dei dati. Le macro sono molto utili per manipolare e validare i dati sulle form. Per esempio, si può definire una macro che reagisce a diversi valori di un attributo con diversi messaggi, e garantire in modo sofisticato che i dati inseriti siano corretti.

- Impostare le proprietà delle form, dei report e dei campi. Con le macro si possono automatizzare i cambiamenti di qualsiasi proprietà di questi oggetti. Ad esempio, è possibile rendere invisibile una form quando serve il suo contenuto ma non serve che questo venga visualizzato.

- Automatizzare i trasferimenti dei dati. Se si devono trasferire ripetutamente dati tra Access ed altre applicazioni (sia in lettura che in scrittura), si può automatizzare il compito.

- Creare un proprio ambiente di lavoro. Si può specificare una macro che apre tutto un insieme di tabelle, query, form e report tutte le volte che si apre un database, personalizzando eventualmente le barre degli strumenti.

- Specificare le reazioni a certi eventi. Per ogni campo di una form è possibile specificare quale macro debba essere eseguita in corrispondenza di ogni evento di accesso, selezione o modifica. Questa caratteristica costituisce la base per la definizione di comportamenti reattivi in Access, che però non devono

essere confusi con le basi di dati attive descritte nel capitolo 12; infatti, tali comportamenti scattano in Access solo quando viene utilizzata una form particolare per la manipolazione della base di dati, mentre non scattano se la stessa operazione è eseguita direttamente in SQL o tramite un'altra form.

Per la definizione di macro bisogna selezionare la voce Macro dalla finestra di base del database e premere quindi il bottone NEW. A questo punto si apre la finestra di progettazione di macro. La finestra contiene nella metà superiore una tabella con due colonne: Action e Comment. La macro è composta da una sequenza di azioni descritte nella colonna Action, ciascuna su una diversa riga, a cui può essere associata una breve descrizione. Nella metà inferiore della tabella compaiono un insieme di attributi che, per ogni azione singolarmente, specificano i parametri dell'azione. È possibile specificare delle condizioni che devono essere verificate affinché un comando venga eseguito, aggiungendo la colonna Condition. Si può inoltre far uso di semplici strutture di controllo.

I comandi disponibili possono essere divisi in tipologie. Vediamo i comandi più significativi di ciascuna tipologia.

- Esecuzione: `DoMenuItem` (esegui un opzione di un menu), `OpenQuery` (esegue una query), `RunMacro` (esegue una macro), `RunSQL` (esegue un comando SQL), `RunApp` (esegue un'applicazione esterna), `CancelEvent` (scarta l'ultimo evento), `Quit` (chiude la sessione), `StopMacro` (termina l'esecuzione della macro) e `StopAllMacros` (termina l'esecuzione di tutte le macro).

- Accesso ai dati: `ApplyFilter` (applica una condizione di selezione alle tuple), `FindNext` (passa alla tupla successiva), `FindRecord` (accede a una tupla),

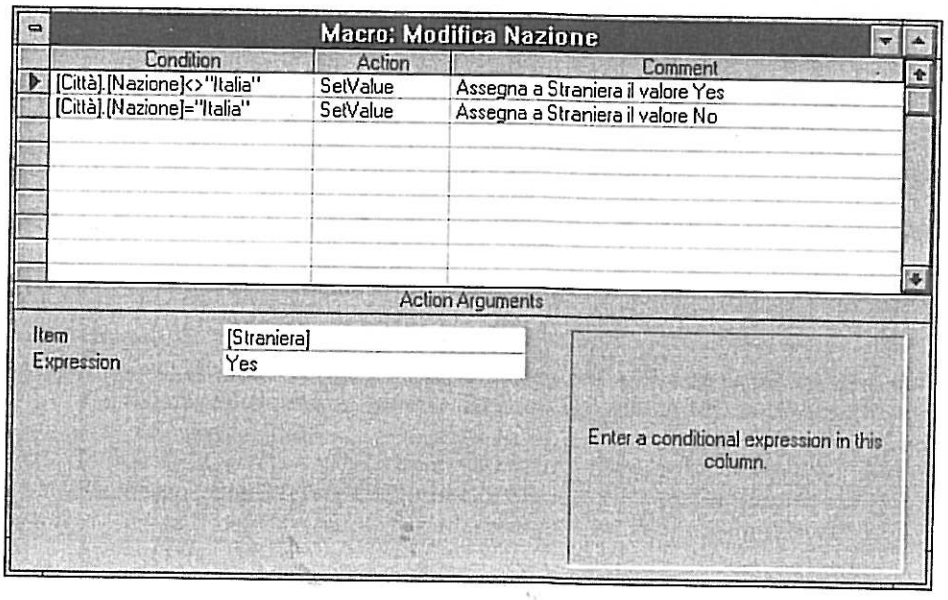

Figura D.20 La finestra di definizione delle macro

`GoToControl` (posiziona il cursore in un control della form), `GoToRecord` (va ad una particolare tupla).

- Modifica dei dati: `SetValue` (assegna a un elemento un particolare valore), `DeleteObject` (cancella un oggetto), `OpenForm`, `OpenQuery`, `OpenReport`, `OpenTable` (comandi che aprono rispettivamente una form, una query, un report e una tabella).

Vi sono poi altre famiglie di comandi, come quelli per trasferire dati tra Access e altre applicazioni, per modificare la dimensione delle finestre e per aprire delle finestre di dialogo con l'utente.

Una semplice macro è descritta in figura D.20. La macro è associata alle modifiche sull'attributo Nazione di una persona; la macro assegna all'attributo Straniero il valore *Yes* se la nazione è diversa dall'Italia, altrimenti assegna all'attributo il valore *No*.