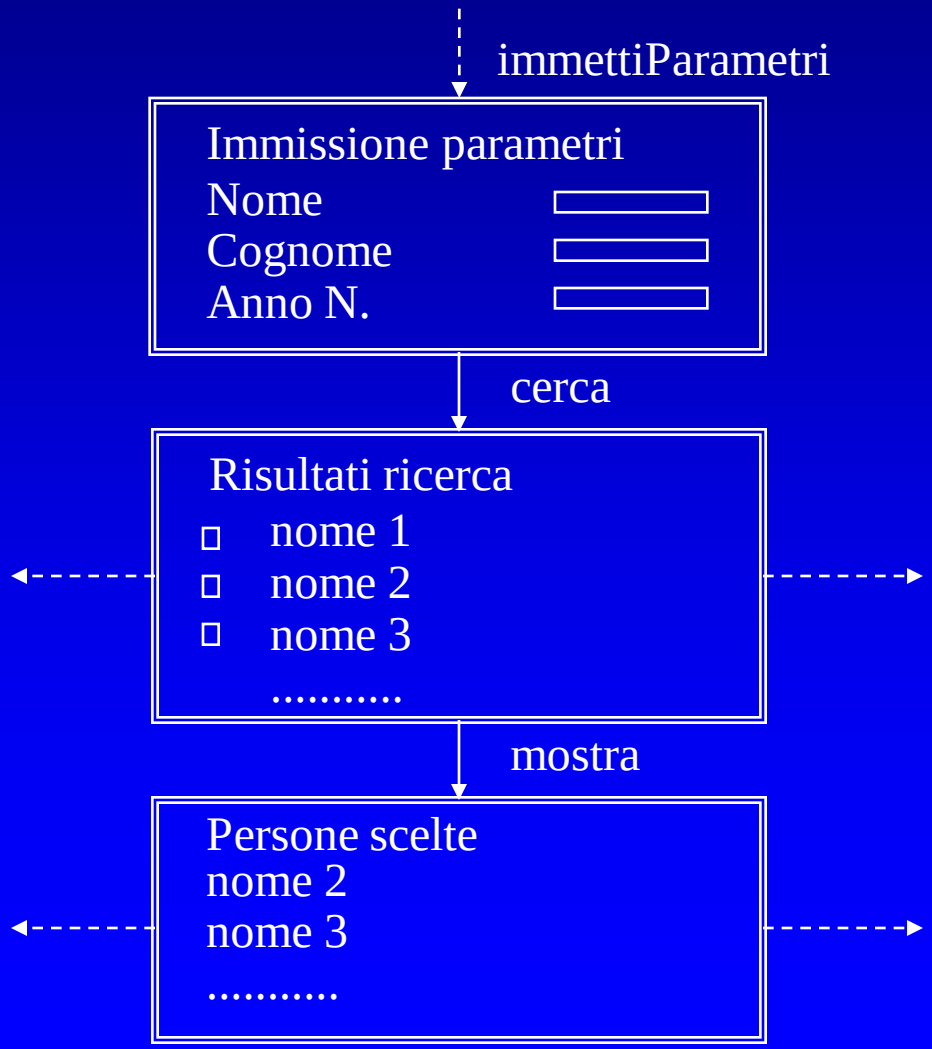
# Esercitazione 3

- Utilizzo di un array per la visualizzazione dei dati

# Operazione ricerca e selezione persone

- Vogliamo implementare il seguente progetto:



# Progetto di dettaglio

- Stato 0
  - Operazione: immettiParametri
  - Parametri: IlNome, IlCognome, LAnnoNascita
- Stato 1
  - Operazione: cerca
  - Parametri: IlNome, IlCognome, LAnnoNascita
- Stato 2
  - Operazione: mostra
  - Parametri: TABLE di VARCHAR2

# Per cominciare

- Creo una directory ~/*ese3bdl*
- Copio il contenuto della directory *~ghelli/bdl/esercizi/ese3* nella directory ~/*ese3bdl*

  oppure visitare la pagina http://www.di.unipi.it/~ghelli/didattica/bdl/bdl.html
- Mi connetto a *oracle* via *sqlDeveloper*

# Per mettersi in pari

- Se non ho già creato una tabella *Persone* nell'ultima lezione:
  - Con sqlDeveloper apro e compilo i file *ese1/create.sql* e *ese1/insert.sql*
- Apro il file *ese2/ese2.pks* (possibilmente con *emacs o wordpad*)
- Sostituisco tutte le occorrenze di XXX e MioAccountOracle con il mio nome nome_utente e salvo.
- Carico *ese2/ese2.pks* ed *ese2/ese2.pkb*

# Per implementare

- Apro e compilo, nell'ordine, i file *modGUI.pks*, *modGUI.pkb*, *ese3.pks* ed *ese3.pkb*
- Verifico se le modifiche sono avvenute controllando i pacchetti P_ESE3 e MODGUI
- Verifico il funzionamento dell'applicazione aprendo un browser all'indirizzo: *http://oracle2.cli.di.unipi.it/pls/mioAccountOracle.p_ese3.immettiParametri*

# Invocare l'applicazione dal Web

- Per invocare la procedura dal Web, modifico il file *~/public_html/ese2bdl/menu.html*, creato alla precedente lezione sul mio spazio web, e aggiungo una form per la chiamata della procedura *MioAccountOracle.p_ese2.immettiParametri*

```
<FORM METHOD="GET"
  ACTION="http://oracle2.cli.di.unipi.it/pls/
  MioAccountOracle.p_ese3.immettiParametri">
<INPUT TYPE= "SUBMIT" VALUE= "Cerca e
  seleziona Persone">
</FORM>
```

# Spedire e ricevere liste di stringhe

- Una lista di stringhe è spedita da una URL:
  - **…/user.pack.proc?a=10&a=20&a=30**
- Il web listener sa che il tipo di **a** in **user.pack.proc** è **TABLE OF VARCHAR2(XXX) INDEX BY BINARY_INTEGER**
- traduce la URL in:

```
TYPE parTable IS TABLE OF VARCHAR2(XXX) INDEX
  BY BINARY_INTEGER;
parTable a;
a(1):=10; a(2):=20; a(3):=30;
user.pack.proc(a);
```

# Spedire liste di stringhe

- Una lista di stringhe è spedita da una URL:
  …**/user.pack.proc?a=10&a=20&a=30**
- Ovvero da una form '**ACTION="user.pack.proc"**' con tanti componenti input che hanno lo stesso nome '**a**':

```
…
<INPUT TYPE="checkbox" NAME="ICognomi"
  VALUE="Rossi">
…
<INPUT TYPE="checkbox" NAME="ICognomi"
  VALUE="Rossani">
…
<INPUT TYPE="checkbox" NAME="ICognomi"
  VALUE="Rossetti">
…
```

# Ricevere liste di stringhe

- Definisco un tipo tabella ed un valore tabella vuota:

```
TYPE HTTPCognomiT IS TABLE OF VARCHAR2(15) INDEX
  BY BINARY_INTEGER;
EmptyHTTPCognomi HTTPCognomiT;
```

- Li uso come tipo parametro e valore default:

```
procedure mostra(
  ICognomi HTTPCognomiT default EmptyHTTPCognomi)
```

# Esercizi

- Dare un messaggio diverso quando la lista è vuota
- Sostituire gli usi di htp… con ModGUI…
- Sostituire la scelta dell'anno di nascita: invece di usare una formText usare una combo box per scegliere tra 1985 e 1995:
  - ApriSelect('LAnnoNascita');
    AggiungiSelect('1985','OttantaCinque');
    …
    ChiudiSelect;
- Lo stesso, ma estraendo gli anni dal BD (senza distinguere stavolta 1985 e 'OttantaCinque')
- Trasferire l'esempio su di una relazione dello schema