ESAME DI GESTIONE DELL'INFORMAZIONE AZIENDALE
III APPELLO

Domanda #1 (3 punti)
Si descrivano (possibilmente per punti) le proprietà/caratteristiche delle chiavi primarie e delle chiavi esterne di una tabella di un Data Base relazionale.

Risposta

Chiave primaria

- È necessaria in ogni tabella
- Ha valore univoco (generalmente numerico, in questo caso può essere anche auto-calcolato)
- Generalmente basata su un solo campo, ma è possibile crearla combinando i valori di più campi (esempio tabella ponte)
- Non può essere nulla

Chiave esterna

- È presente nelle tabelle "figlio" di relazioni OTM, o in tabelle in cui esiste una SRR. Nel caso di una relazione OTO la relazione è definita direttamente sulla chiave primaria. In questo caso chiave primaria e chiave esterna coincidono.
- Può essere nulla
- Deve avere lo stesso tipo di dati della chiave primaria a cui fa riferimento
- Sono ammessi solo valori già presenti tra quelli assunti dalla chiave primaria a cui fa riferimento

Domanda #2 (3 punti)
Spiegare, sinteticamente, cosa s'intende per paradigma ERP.

Risposta
Un gestionale ERP è un sistema informativo di tipo operazionale basato su data base relazionale. Come tutti i sistemi informativi ha l'obiettivo di registrare in maniera sicura ed efficiente le transazioni legate ai processi aziendali, di assicurare un corretto reperimento d'informazione di tipo operativo (informazione che deve essere precisa, accurata, tempestiva, comprensibile) oltre che automatizzare o quantomeno efficientare i processi operativi.
Oltre a questo, un sistema ERP si differenzia da un gestionale aziendale di tipo 'custom' o 'legacy' in quanto implementa tre principi che vanno sotto il nome di paradigma ERP. Tali principi sono:

- Unicità della base di dati: tutte le transazioni operano su uno stesso data base relazionale)
- Modularità: gruppi di processi omogenei (che tipicamente fanno capo ad una specifica funzione aziendale) vengono gestiti con specifici applicativi (es. produzione, manutenzione, logistica e magazzini, ecc.).
- Prescrittività: il sistema impone un modo unico di operare. I processi e le procedure implementate devono necessariamente essere rispettati. Non è possibile bypassare le sequenze di attività/step codificate nel sistema.

Si noti che, unitamente, l'unicità della base di dati e la modularità permettono d'implementare una gestione per processi oltre che uno sviluppo progressivo del sistema ERP.

Domanda #3 (4 punti)
Si consideri la seguente query SQL, contenente una Dfunction ed operante sulla tabella STUDENTS di seguito riportata.

```
SELECT DAVG ("AGE", "STUDENTS") AS AVG_AGE
FROM STUDENTS
WHERE SURNAME LIKE '[AC][!a-e]*'
```

Tabella STUDENTS

ID	SURNAME	AGE	NATIONALITY
1	Brixell	25	AMERICAN
2	Azady	24	IRANIAN
3	Cromwell	18	BRITISH
4	Abbadi	22	IRANIAN
5	Crowford	25	AMERICAN
6	Hishimury	25	JAPANESE
7	Babbell	21	BRITISH

Si chiede di:

- Disegnare la tabella restituita dalla query,
- Commentare il risultato ottenuto (Max 3-4 righe).

Soluzione

AVG_AGE
22.85
22.85
22.85

La DFunction, essendo una funzione VBA, viene eseguita solo in un secondo momento. Prima viene eseguita la query SQL priva della DFunction. Tale query restituisce tre record uno per ciascuno degli studenti il cui cognome rispetta la condizione `LIKE '[AC][!a-e]*`. Nella fattispecie tali studenti sono Azady, Cromwell e Crowford. A questo punto per ogni record viene calcolata la DAVG che calcola la media complessiva (non è infatti presente alcun filtro nella DAVG). La media vale allora (25 + 24 + 18 + … +21) /7 = 22,857. Tale valore viene quindi restituito tre volte, uno per ogni record restituito dalla query originaria, come mostrato in tabella.

Domanda #4 (5 punti)

Si consideri il seguente codice VBA operante sulla tabella STUDENTS mostrata alla domanda #3:

```
Dim Rcs As Recordset2
Dim MySQL As String
Dim N As Integer
Dim M As Double
MySQL = "SELECT SURNAME, AGE FROM STUDENTS WHERE NATIONALITY IN ('IRANIAN', 'JAPANESE')"
Set Rcs = CurrentDb.Openrecordset(MySQL)
IF Rcs.RecordCount <= 0 Then
        Debug.Print "Empty DataBase"
Else
        Debug.Print Rcs.Fields(0) & "-" & Rcs.Fields(1)
        Rcs.FindLast ("Age <= 24")
        Do While Not Rcs.BOF    'BOF means Beginning of File
                N = N + 1
                M = M + Rcs.Fields(1)
                Rcs.MovePrevious
        Loop
        X  =  (N*M)
        Debug.Print Rcs.Fields(0) & "-" & Rcs.Fields(1)
        Debug.Print "X = " & Cstr(X)
End IF
…
```

Si provveda a:

- Scrivere (su foglio) ciò che comparirebbe a video (o meglio nella finestra immediata) eseguendo tale codice. A tal fine si ricorda che i valori mostrati a video sono quelli relativi alle istruzioni Debug.Print, evidenziate in grassetto nel codice.
- Spiegarne, sinteticamente il principio di funzionamento.

Soluzione

Viene aperto un recordset su una query che filtra la tabella studenti per nazione. Qualora non ci fosse nessuno studente che soddisfa il criterio sulla nazione di origine il codice si arresterebbe subito, il recordset sarebbe vuoto, il metodo recordcount restituirebbe 0 e, a video si leggerebbe la scritta "empty database".

Nel caso d'esempio però, alcuni record soddisfano il criterio e il recordset contiene i tre record mostrati nella tabella seguente.

SURNAME	AGE
Azady	24
Abbadi	22
Hishimury	25

All'apertura il cursore (di riga) è posizionato sul primo record, per questo, l'istruzione *Debug.Print Rcs.Fields(0) & "-" & Rcs.Fields(1)* restituisce a video la seguente scritta: Azady-24.

A questo punto il cursore viene spostato sull'ultimo record (Rcs.FindLast ("Age <= 24")) corrispondente ad uno studente di età minore uguale a 24. Come si vede in tabella, si tratta del secondo record.
Il codice a questo punto esplora a ritroso (risalendo dal basso verso l'alto) tutti i record del recordset.
Alla prima iterazione:

- *N = N + 1 → N = 1*
- *M = M + Rcs.Fields(1) → M = 22*

Alla seconda iterazione:

- *N = N + 1 → N =1 + 1 = 2*
- *M = M + Rcs.Fields(1) → M = 22 + 24 = 46*

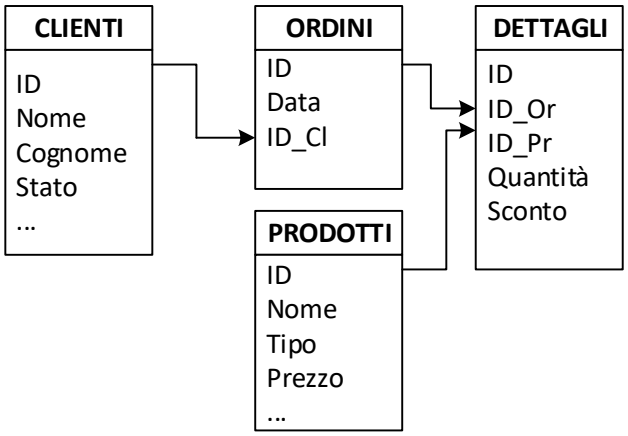
All'uscita del ciclo, quindi la variabile X vale 46*2 = 92
A video verrà allora mostrato:

- Azady-24
- X = 92

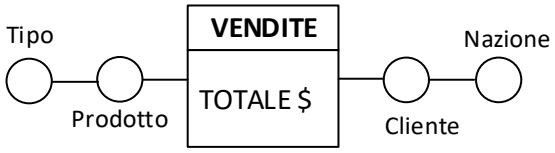
L'intera lista delle scritte che appaiono a video è allora la seguente:

- Azady-24
- Azady-24
- X = 92

Esercizio #1. (9 Punti)
Si considerino le quattro tabelle seguenti, relative al processo di acquisto di un cliente.



Si supponga che tali tabelle alimentino un Data Warehouse, la cui struttura concettuale sia definita dal seguente diagramma DFM.



Si chiede di:

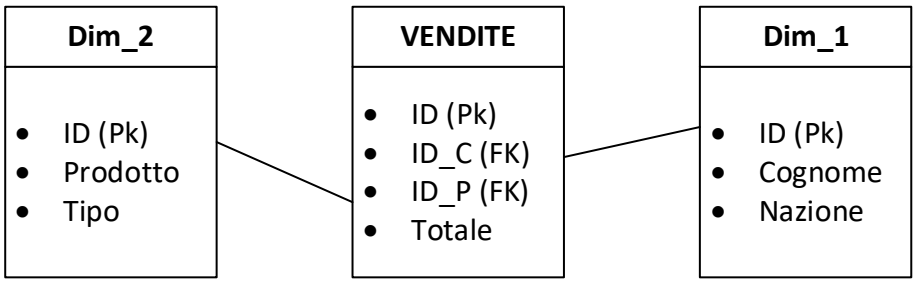
- Disegnare lo schema entità-relazioni delle tabelle dei fatti e delle dimensioni che formano il data Warehouse, nell'ipotesi che esso sia realizzato tramite architettura ROLAP a stella.

- Scrivere la query SQL che, operando sul ROLAP prima definito, realizza la seguente query multi dimensionale: VENDITE(Stato = USA, *).$
- Compilare i record della tabella dei fatti e delle tabelle delle dimensioni, supponendo che la vista di partenza (volgarmente il “tabellone iniziale”) sia la seguente:

Cognome	Stato	Prodotto	Tipo	Totale $
Zurli	ITALIA	SL_AAA	Semi Lavorato	47
Zurli	ITALIA	PF_AAA	Prodotto Finito	36
Rossi	ITALIA	SL_AAA	Semi Lavorato	82
Wiley	USA	PF_BBB	Prodotto Finito	80
Wiley	USA	PF_AAA	Prodotto Finito	120

Soluzione

Il fatto di vendita ha una sola misura ed è descritto secondo due dimensioni, ciascuna avente una gerarchia a due livelli. Lo schema è allora il seguente:



Operando la seguente query multidimensionale VENDITE(Stato = USA, *).$ vogliamo le vendite di tutti i prodotti effettuate in USA. Non essendo definita una specifica funzione di aggregazione useremo la somma (aggregazione additiva). La query SQL è allora la seguente:

```
SELECT Sum(Vendite.Totale)
FROM Dim2 Inner Join (Vendite Inner Join Dim_1 On Vendite.ID_C = Dim_1.ID) On
                                                    Dim_2.ID = Vendite.ID_P
WHERE Dim_1.Nazione = ‘USA’
```

Le tre tabelle (compilate) sono le seguenti:

Tabella DIM_1

ID	Cognome	Nazione
1	Zurli	Italia
2	Rossi	Italia
3	Wiley	USA

Tabella DIM_2

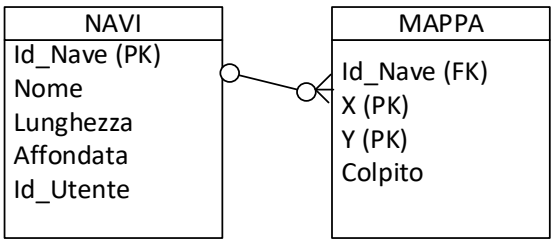
ID	Prodotto	Tipo
1	SL_AAA	Semi Lavorato
2	PF_AAA	Prodotto Finito
3	PF_BBB	Prodotto Finito

Tabella Fatti di Vendita

ID	ID_P	ID_C	Totale $
1	1	1	47
2	2	1	36
3	1	2	82
4	3	3	80
5	2	3	120

Esercizio #2. (9 punti)

Si considerino due tabelle NAVI e MAPPA, facenti parte di un ipotetico software, che riproduce il gioco della battaglia navale. Come mostrato in figura, la tabella NAVI definisce la tipologia di navi, la tabella MAPPA la loro disposizione.



Ad esempio, le due navi in figura (supposte appartenenti alla flotta dell'utente U1), di cui una è già stata colpita, sarebbero codificate in questo modo (la cella 0,0 è l'origine convenzionale):

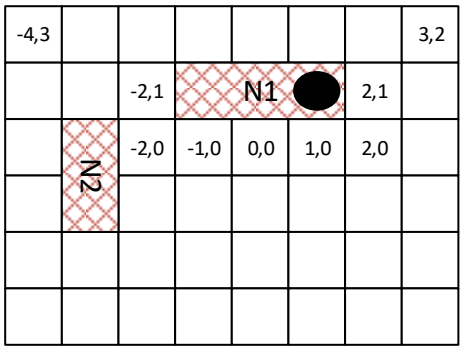


Tabella Navi

Id_Nave	Nome	Lunghezza	Id_Utente	Affondata
N1	Fregata	3	U1	False
N2	Torpediniere	2	U1	False

Tabella Mappa

ID	Id_Nave	X	Y	Colpita
1	N1	-1	1	False
2	N1	0	1	False
3	N1	1	1	True
4	N2	-3	0	False
5	N2	-3	-2	False

Si chiede di scrivere:

- Una funzione Privata "Shoot": *Private Function Shoot(X As Integer, Y As Integer, ID_Opponent As String) As Integer*
  Shoot riceve in input tre valori interi (due corrispondenti alle coordinate cartesiane della mappa ed uno all'ID dell'avversario) e controlla nella tabella MAPPA la presenza delle coordinate fornite, limitatamente alle navi dell'avversario. Se il colpo è andato a vuoto (in mare) Shoot restituisce 0, se il colpo è andato a segno Shoot restituisce 1 e se il colpo ha affondato la nave (ossia con esso tutti i punti della nave sono stati colpiti) Shoot restituisce 2. A seguito dell'esito dello sparo, la funzione Shoot provvede anche ad aggiornare correttamente i campi "Colpita" e "Affondata", delle tabelle NAVI e MAPPA.

- Una funzione privata To_Sink: *Private Function To_Sink(ID As Integer) As Integer*
  To_Sink riceve in input l'ID di una nave e restituisce in output in numero minimo di colpi che sono ancora necessari per affondarla (ossia lunghezza – numero di punti della nave che sono stati colpiti). Si assuma che l'ID passato in input sia sempre corretto.

Soluzione

```
Private Function Shoot(X As Integer, Y As Integer, ID_Opponent As String) As Integer
Dim Rcs As Recordset2
Dim ID_Nave As String
Dim SQL As String
'Apriamo un recordset sulla tabella MAPPA considerando solo le navi dell'avversario che non sono affondate e le coordinate non ancora colpite. Filtriamo inoltre per X e Y, se viene restituito un record allora abbiamo colpito la nave, altrimenti colpo a mare'
        SQL = "SELECT * FROM MAPPA INNER JOIN NAVI ON MAPPA.ID_Nave = NAVI.ID_Nave"
        SQL = SQL & " WHERE NAVI.Affondata = False AND MAPPA.COLPITA = False AND"
        SQL = SQL & " NAVI.ID_Utente = '" & ID_Opponent & "'"
        SQL = SQL & " AND X = " & X  & " AND Y = " & Y
        Set Rcs = CurrentDb.Openrecordset(SQL)
        If Rcs.EOF Then
                Shoot = 0
        Else 'Abbiamo colpito, salviamo l'ID della nave colpita e registriamo il colpo
                ID_Nave = Rcs.Fields!ID_Nave
                Shoot = 1
                Rcs.Edit
                Rcs.Fields!Colpita = True
                Rcs.Update
                'Verifichiamo che la nave sia stata affondata. In Questo caso aggiorniamo
                If Nave_Affondata(ID_Nave) Then:
                        Shoot = 2
                        Call Aggiorna(ID_Nave)
                End If
        End If
        Rcs.Close
        Set Rcs = Nothing
End Function

Private Function Nave_Affondata(ID As String) As Boolean
'Per vedere se la nave è affondata usiamo to la funzione to_sink che restituisce il numero di colpi
'mancanti per l'affondamento
Dim Mancanti As Integer
        Mancanti = To_Sink(ID)
        If Mancanti = 0 Then
                Nave_Affondata = True
        Else
                Nave_Affondata = False
        End If
End Function
```

```
Private Sub Aggiorna(ID As String)
Dim MySQL As String
        MySQL = “UPDATE NAVI  SET Affondata = True WHERE ID_Nave = ” & ID
        Application.DoCmd.RunQuery(MySQL)
End Sub

Private Function To_Sink(ID As String) As Integer
‘Confrontiamo la lunghezza della nave con il numero di colpi subiti dalla nave.
‘A tal proposito usiamo due DFunction
Dim L As Integer, N As Integer
Dim Condition As String
        L = DlookUp(“Lunghezza”, “NAVI”, “ID_Nave = ” & ID)
        Condition = “ID_Nave = ” & ID & “ AND Colpita = True)
        N = DCount(“ID_Nave”, “MAPPA”, Condition)
        To_Sink = L - N
End Function
```