PREAPPELLO DI GESTIONE DELL'INFORMAZIONE AZIENDALE
20/12/2021

SEZIONE 1 – DOMANDE A SCELTA MULTIPLA

Domanda 1.
Scegliere la risposta corretta:

- ☐ Una tabella deve avere una chiave esterna,
- ☐ Una tabella può avere una o più chiavi esterne,
- ☐ Una tabella può avere al più una chiave esterna,
- ☐ Una tabella può avere una chiave primaria,
- ☐ Nessuna delle precedenti risposte è corretta.

Domanda 2.
Scegliere la risposta corretta, relativamente ad un sistema MOLAP:

- ☐ È una tipologia di Data Warehouse,
- ☐ È un Datawarehouse implementato mediante un Database relazionale,
- ☐ È un fatto multidimensionale,
- ☐ È una matrice multidimensionale,
- ☐ Nessuna delle precedenti risposte è corretta.

Domanda 3.
Si indichi la stringa che non soddisfa la seguente condizione LIKE '###[!AB]*[A-Z]'

- ☐ 123CCCCCC
- ☐ 123_123Z
- ☐ 123_Z
- ☐ 123CCC1
- ☐ 123_CDE

Domanda 4
Un gestionale ERP

- ☐ È un sistema di Business Intelligence,
- ☐ È spesso utilizzato dal top management aziendale,
- ☐ Abbraccia i processi di un'intera supply chain,
- ☐ Rispetta i criteri FASMI,
- ☐ Nessuna delle precedenti risposte è corrette.

Domanda 5
Sia X una variabile Stringa, T la tabella Anagrafica Clienti (con numero di record maggiore di zero), ID la chiave primaria, numerica ed autocalcolata. Allora la seguente istruzione X = Nz(DLookup("Cognome", "T", "ID = 0"), -1) restituisce:

- ☐ Sicuramente il valore "-1",
- ☐ Il cognome dell'utente con ID = 0,
- ☐ Il primo cognome trovato, tra quello delle persone con ID = 0,
- ☐ Si genera un errore perché l'ID non può essere nullo o uguale a zero,
- ☐ Si genera un errore perché X è di tipo stringa.

Domanda 6

Si consideri la seguente funzione.

```
Public Function foo(X As Variant) As Variant
        foo = rnd()*100
        foo = X
End Function
```

Che valore assume Y se venisse fatto il seguente assegnamento Y = foo(Y)

- ☐ La funzione dà errore perché invocata con Y invece che con X,
- ☐ Il valore di Y è random, ma sicuramente minore di 100,
- ☐ Il valore di Y non cambia, e non cambia il suo tipo,
- ☐ Il valore di Y non cambia, ma Y diventa di tipo variant,
- ☐ Nessuna delle precedenti risposte è corretta.

Domanda 7

Si considerino le seguenti istruzioni:

```
Dim B As Boolean
Dim i As Integer
i = 0
B = True
Do While B
        i = i + 2
        if i = 3 Then Exit Do
Loop
```

- ☐ Il ciclo non parte perché B è già vero,
- ☐ Il ciclo va avanti all'infinito perché B è sempre vero,
- ☐ Il ciclo s'interrompe quando i diventa pari a 3,
- ☐ Il ciclo va avanti all'infinito perché i non può assumere il valore di 3
- ☐ Nessuna delle precedenti risposte è corretta.

Domanda 8

Si scelga l'affermazione corretta circa una relazione OTO.

- ☐ È un caso particolare di relazione Many To Many,
- ☐ È un caso particolare di relazione One To Many fatta tra chiavi primarie,
- ☐ È utilizzata per ricreare una gerarchia,
- ☐ È usata tipicamente quando tutti i dati di una tabella sono strutturati
- ☐ Nessuna delle precedenti risposte è corretta.

Domanda 9

L'operazione di JOIN di SQL può essere fatta:

- ☐ Esclusivamente tra tabelle in relazione OTM,
- ☐ Esclusivamente tra tabelle, anche se non relazionate tra di loro,
- ☐ Anche tra viste e/o tra viste e tabelle,
- ☐ Anche tra i campi di un record,
- ☐ Nessuna delle precedenti risposte è corretta.

Domanda 10

Il prodotto cartesiano fatto tra la tabella A con 3 record di 3 campi e la tabella B con 5 record di 5 campi

- ☐ Non può essere fatto perché A ha meno campi di B,
- ☐ Non può essere fatto perché A ha meno record di B,
- ☐ Avrà 3*5 record, ciascuno con 3*5 campi
- ☐ Avrà 3^5 record con 3*5 campi
- ☐ Avrà 3*5 record, ciascuno con 3+5 campi

SEZIONE 2 – DOMANDE ED ESERCIZI

Domanda 1 (4 punti)

"Un Data Base Relazionale ha una struttura orientata ai processi ed alle transazioni, mentre un Data Warehouse ha una struttura orientata ai fatti ed agli eventi". Tale affermazione può ritenersi corretta? Si spieghi.

Risposta

L'affermazione è corretta. Un Data Base Relazionale, di un SI operazionale ha una struttura che: (i) ricalca i processi che generano le transazioni (che verranno archiviate nel DB) e che (ii) include (a livello di tabelle ed anagrafiche) le entità che partecipano a tali processi.

L'obiettivo è ottimizzare l'archiviazione dei dati, cosa che comporta una parziale frammentazione dell'informazione che, quindi, deve essere ricostruita tramite query di SELECT.

Per ovviare a tale fatto, i dati di un data Warehouse sono riorganizzati secondo un approccio multidimensionale, attorno ai "fatti" a cui essi partecipano. Ciascun fatto definisce un evento/fatto d'interesse aziendale rappresentato secondo un certo numero di dimensioni (ed eventuali gerarchie), e di metriche (misure che quantificano il fatto)

Es. Processo di manutenzione:

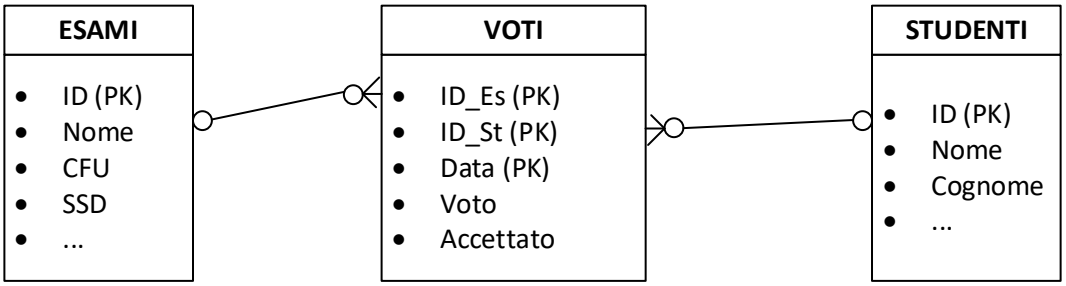
- Si registra la richiesta d'intervento (in cui si indica il tipo di malfunzionamento, la macchina e il reparto interessato, ecc.), la richiesta è presa a carico dal responsabile di manutenzione che genera un ordine di manutenzione (OdM) assegnando tempi e risorse. I manutentori eseguono l'intervento registrando tipo di guasto, tempi, materiali e componenti sostituiti, ecc.

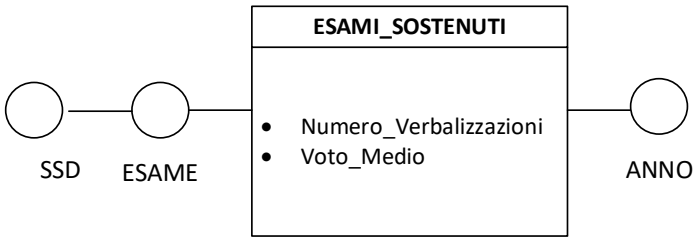
Es. Fatto di manutenzione

- Dimensioni: 1) Componente – Macchina -Reparto, 2) Manutentore 3) Guasto – Tipo Guasto
- Metriche: 1) Tempo di fermo, 2) Tempo di riparazione 3) Numero componenti cambiati …

Domanda 2 (6 punti)

Si consideri il seguente schema Entità Relazioni (SSD sta per Settore Scientifico e Disciplinare) ed il seguente DFM.





- Se ne spieghi brevemente il significato,
- Si scriva la query per la creazione della vista (tabellone) iniziale,
- Supponendo di usare un ROLAP, si scrivano le query di popolamento delle tabelle delle dimensioni, e quella di popolamento della tabella del fatto.

Risposta

Il primo schema rappresenta il diagramma entità relazioni della parte di un DBR utilizzato per la prenotazione agli esami e per la registrazione dei voti conseguiti.
Le due anagrafiche (esami e studenti) sono in relazione MTM e la tabella VOTI funge da tabella ponte. Tale tabella ha 3 campi usati come chiave primaria, le due FK e la data dell'esame (tripletta di dati che assicura l'unicità). Voto e accettato sono due campi inizialmente nulli (al momento dell'iscrizione) e verranno successivamente compilati dall'insegnante (a fine correzione) e, eventualmente, dallo studente dopo la visualizzazione di compito e voto conseguito.

Il secondo schema è un DFM che mostra un fatto a due dimensioni generato a partire dallo schema relazionale prima definito.
In particolare, si hanno due dimensioni (dimensione temporale e dimensione dell'esame), la seconda dotata di una gerarchia. Ogni fatto rappresenta l'insieme degli esami (di una data materia) sostenuti in un certo anno. Ogni fatto è quantificato o come numero di "promossi" (voto superiore a 18 accettato) e come voto medio conseguito. Data la presenza della gerarchia è possibile aggregare gli esami per settore, ottenendo così il numero di verbalizzazioni e il voto medio per settore disciplinare e per anno.

La query che restituisce la vista iniziale deve contenere un campo per ogni dimensione/gerarchia e per ogni metrica. Per cui, provvedendo anche a raggruppare secondo il livello di granularità richiesto dal fatto, si ha:

```
SELECT ESAMI.Nome, ESAMI.SSD, YEAR(VOTI.Data) As Anno, AVG(VOTI.Voto) As Voto_Medio
        COUNT(VOTI.ACCETTATO) As Numero_Verbalizzazioni
FROM ESAMI INNER JOIN VOTI ON ESAMI.ID = VOTI.ID_Es
WHERE ACCETTATO = TRUE
GROUP BY ESAMI.Nome, ESAMI.SSD, YEAR(VOTI.Data) As Anno
```

Si noti che abbiamo assunto che la media si riferisca ai soli voti accettati (in assenza di tale specificazione anche la media generale può comunque ritenersi accettabile)

Le tabelle delle dimensioni sono

- ESAMI {ID (PK), NOME; SSD}
- TEMPO {ID (PF), ANNO}

Si tratta di inserire in tali tabelle tutte le possibili coordinate. Allora:

```
INSERT INTO ESAMI (Nome, SSD)
SELECT DISTINCT Nome, SSD
FROM VISTA_INIZIALE

INSERT INTO TEMPO (Anno)
SELECT DISTINCT Anno
FROM VISTA_INIZIALE
```

Dove abbiamo indicato con VISTA_INIZIALE il nome del “tabellone di apertura”

Infine, la tabella del fatto sarà:

- FATTO_ESAMI {ID (PK), ID_Es(FK), ID_An(FK), Voto_Medio, N_Verbalizzazioni}

Per compilare le due FK serve una JOIN tra la vista iniziale e le tabelle delle dimensioni appena create. Per cui:

```
INSERT INTO FATTO_ESAMI( ID_Es, ID_An, Voto_Medio, N_Verbalizzazioni)
SELECT ID_Es, ID_An, Voto_Medio, N_Verbalizzazioni
FROM ESAMI, TEMPO, VISTA_INIZIALE
WHERE ESAMI.NOME = VISTA_INIZIALE.Nome, TEMPO.ANNO = VISTA_INIZIALE.ANNO
```

Domanda 3 (4)

Si consideri la tabella VOTI dello schema precedente. Si riscriva la seguente procedura usando un recordset.

```
Public Sub Voto(ID_St As Integer, ID_Es As Integer, D As Date, V As Integer)
Dim MySQL As String, Wh_Cond As String
        MySQL = “UPDATE VOTI SET VOTO = ” & V
        Wh_Cond = “ WHERE ID_St = ” & ID_St & “ AND ID_Es = ” & ID_Es & “ And Data = # ” & D & “#”
        DoCmd.RunSQL(MySQL & Wh_Cond)
End Sub
```

Risposta

La subroutine esegue una query di update, che permette di inserire il voto relativo conseguito da uno studente (per un certo esame sostenuto in una certa data)

Usando un recordset invece che una query di update possiamo riscrivere la procedura nel modo seguente.

```
Public Sub Voto(ID_St As Integer, ID_Es As Integer, D As Date, V As Integer)
Dim Wh_Cond As String
Dim Rcs As Recordset2
        Set Rcs = CurrentDb.OpenRecordset("VOTI", dbOpenDynaset)
        Wh_Cond = "ID_St = " & ID_St & " And ID_Es = " & ID_Es & " And Data = #" & D & "#"
        Rcs.FindFirst (Wh_Cond)
        If Not Rcs.NoMatch Then 'Se non mettessi tale condizione, se non trovassi il record cercato, aggiornerei il primo
                Rcs.Edit
                Rcs.Fields("Voto") = V
                Rcs.Update
        End If
   Rcs.Close
   Set Rcs = Nothing
End Sub
```

Domanda 4 (6 punti + 3 facoltativo)

Si consideri la seguente funzione ricorsiva.

```
Public Function Foo(Vals() As Integer, Optional Pos As Variant) As Long
   'Alla prima chiamata Pos non viene passato
   If IsMissing(Pos) Then Pos = LBound(Vals) 'sarà zero o uno in funzione dell’indicizzazione di Vals
   If Pos = UBound(Vals) Then 'Condizione di uscita
      Foo = Vals(Pos)
   Else
      Foo = Vals(Pos) * Foo(Vals, Pos + 1) 'Chiamata ricorsiva
   End If
End Function
```

Si chiede di:

- Descriverne il funzionamento.
- Indicare il vettore che andrebbe passato in input per ottenere, in uscita il valore di 5!
- Disegnare lo schema della *"pila di stack"* che mostra le chiamate ricorsive e i valori scambiati tra le funzioni, nel caso in cui il vettore di input sia {3,4,5}.

Risposta

La funzione esegue la produttoria degli elementi di un vettore passato in input. Alla prima chiamata il parametro opzionale pos prende il valore del primo indice del vettore (0 o 1 in funzione dell'indicizzazione del vettore stesso). La funzione restituisce in output il valore in posizione pos moltiplicato per ciò che restituisce la stessa funzione chiamata però con pos = pos + 1 (ossia il valore successivo).
In pratica quindi, ciascuna funzione si limita a moltiplicare il valore in posizione pos con ciò che verrà restituito dalle chiamate successive. In questo modo si ottiene, appunto, la produttoria degli elementi del vettore

Il vettore passato in input dovrebbe essere {5,4,3,2} oppure {5,4,3,2,1}

Ipotizzando un'indicizzazione a base zero si avrebbe:

- Foo({3,4,5}) = 3 * Foo({3,4,5}, 1}
    - Foo({3,4,5}, 1} = 4 * Foo({3,4,5},2}
        - Foo({3,4,5},2} = 5
            - End

Procedendo dal basso verso l'alto, i valori di ritorno sono:

- La terza ed ultima funzione restituisce 5,
- 5 viene moltiplicato per 4, per cui la seconda funzione restituisce 20
- Infine, 20 viene moltiplicato per 3 e la funzione di partenza restituisce 60

Facoltativo

Si riscriva la stessa funzione usando un approccio iterativo. Quale dei due approcci è migliore in termini di efficienza? Si spieghi.

Si tratta di ciclare sul vettore facendo la produttoria dei suoi valori. Per cui:

```
Public Function Foo2(Vals() As Integer) As Long
Dim X As Long
  X = Vals(LBound(Vals))
  For i = LBound(Vals) + 1 To UBound(Vals)
    X = X * Vals(i)
  Next i
```

```
    Foo2 = X
End Function
```

La procedura iterativa è sicuramente più efficiente, dato che non richiede la creazione dello stack di chiamata (eseguire un ciclo è molto più rapido che aprire e tenere in memoria delle funzioni)