ESAME DI GESTIONE DELL'INFORMAZIONE AZIENDALE
LAUREA MAGISTRALE IN ING. GESTIONALE

Data 21/12/2018
Tempo a disposizione 2 ore
Punteggio minimo per accedere alla prova orale pari a 15/30

Nome _____ Cognome _____

Sezione I - Domande a risposta multipla.

Per le soluzioni si rimanda alla teoria. In caso di dubbio si contatti il docente durante l'orario di ricevimento.

Sezione II - Domande aperte ed esercizi.

Domanda #1 – 3 punti

Compito A - Si illustrino le caratteristiche di un sistema informativo di tipo operazionale, specificando cosa sottintende l'acronimo ACID.
Compito B - Si illustrino le acaratteristiche di un sistema informativo di tipo informazionale, specificando cosa sottintende l'acronimo FASMI (3 punti)

Per le soluzioni si rimanda alla teoria. In caso di dubbio si contatti il docente durante l'orario di ricevimento.

Domanda #2 – 4 punti

Compito A - Si creino e si compilino due semplici tabelle e le si utilizzano per spiegare la sequenza di operazioni di algebra relazionale necessarie ad ottenere una LEFT OUTER JOIN (4 punti).

Compito B - Si creino e si compilino due semplici tabelle e le si utilizzano per spiegare la sequenza di operazioni di algebra relazionale necessarie ad ottenere un RIGHT OUTER JOIN (4 punti)

Soluzione

Consideriamo la Tabella A e la tabella B mostrate di seguito che, per generalità, non sono legate da una relazione OTM e non hanno né chiavi primarie né chiavi esterne (sono tabelle generali, non tabelle di un DBR)

Tabella A

A1	A2	A3
1	A	AA
2	B	BB
3	C	CC

Tabella B

B1	B2
1	X
2	XX
2	XXX
Null	XXXX

Supponiamo ora di volere fare un INNER JOIN tra A e B usando come campi di join il campo A1 della tabella A e il campo B1 della tabella B (campi tra loro compatibili).
Per prima cosa facciamo il prodotto cartesiano. Si ottiene una tabella con 12 record ciascuno contenente 5 campi:

Tabella AxB

A1	A2	A3	B1	B2
1	A	AA	1	X
2	B	BB	1	X
3	C	CC	1	X
1	A	AA	2	XX
2	B	BB	2	XX
3	C	CC	2	XX
1	A	AA	2	XXX
2	B	BB	2	XXX
3	C	CC	2	XXX
1	A	AA	Null	XXXX
2	B	BB	Null	XXXX
3	C	CC	Null	Null

A questo punto effettuiamo una selezione usando come condizione l'uguaglianza dei campi di join della tabella AxB. Inoltre, facendo una proiezione per eliminare un campo di join otteniamo la seguente tabella di INNER JOIN:

Tabella INNER JOIN A - B

A1	A2	A3	B2
1	A	AA	X
2	B	BB	XX
2	B	BB	XXX

Dato che ci sono campi disaccoppiati in entrambe le tabelle ha senso proseguire facendo un Right Join ed un Left Join. In primo restituirà i record di A che non hanno corrispondenza nella tabella B, il secondo quelli di B che non hanno corrispondenza nella tabella A.

Per trovare i campi disaccoppiati di A dobbiamo generare la lista dei valori non ripetuti del campo B1 della tabella B. Per farlo effettuiamo una proiezione su B mantenendo il solo campo B1 ed eliminiamo i duplicati. Chiamiamo la lista ottenuta L1. Facciamo la stessa operazione su A per ottenere la lista distinta del campo A1. Chiamiamo L2 tale lista. A questo punto sottraendo L1 a L2 troviamo solo i valori di A1 che non figurano in B1. Nel nostro caso l'unico valore è il 3. Usiamo i valori di A1 che non figurano in B1 per filtrare la tabella A (selezione). Nel nostro caso resta il campo {3 C CC}. Per concludere dobbiamo fare il prodotto cartesiano di tale record con un record tutto nullo della tabella B2 {null null}, si ottiene:
{3 C CC Null Null}. Ora possiamo unire tale record alla tabella dell'INNER JOIN per ottenere un RIGHT JOIN, come mostrato di seguito (in cui abbiamo tolto il primo valore nullo corrispondente al campo di Join B1 che era stato precedentemente eliminato):

Tabella RIGHT JOIN A - B

A1	A2	A3	B2
1	A	AA	X
2	B	BB	XX
2	B	BB	XXX
3	C	CC	Null

Per ottenere un LEFT JOIN dobbiamo operare in maniera duale, per ottenere i record non accoppiati della tabella B. In questo caso è sufficiente filtrare cercando i record per i quali il campo di Join B1 è nullo. Si ottiene l'unico record {Null XXXX}. Come prima, effettuando un prodotto cartesiano tra questo record ed uno tutto nullo della tabella A ed unendo il risultato alla tabella dell'INNER JOIN si ottiene la tabella di LEFT JOIN mostrata di seguito:

Tabella LEFT JOIN A - B

A1	A2	A3	B2
1	A	AA	X
2	B	BB	XX
2	B	BB	XXX
Null	Null	CC	Null

Domanda #3 – 5 punti

Si consideri la seguente distinta base di una penna a sfera (di tipo Bic©).

- Penna Tipo Bic
  - Assieme cartuccia
    - Cilindro di plastica
    - Cartuccia
      - *Cannuccia vuota*
      - *Inchiostro*
      - *Punta a sfera*

- Tappo
- Tappino

Si completi la distinta base, inserendo i parametri mancanti ritenuti fondamentali, e la si codifichi in un Data Base relazionale.

Compito A - A tal fine si usi una struttura a 3 tabelle che fa uso di una SRR (3 punti)

Compito B – A tal fime si usi una struttura a 2 tabelle basata su tecnica di Aliasing (3 punti)

Si scriva una query parametrica che riceve in input il nome di una penna (end product) e restituisce tutti i componenti di primo livello di cui essa è composta, specificandone la quantità necessaria (2 punti)

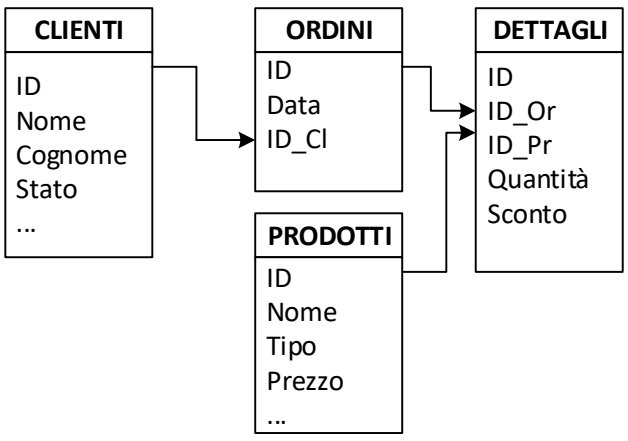
SOLUZIONE

Per la soluzione si faccia riferimento al file allegato (nel quale le suddette strutture a 2 e a 3 tabelle sono visualizzabili mediante il menù "relazioni" del Data Base). Si precisa che i dati inseriti nel DB non fanno riferimento solo alla penna biro BIC, ma anche ad una versione Plus dotata di azionamento a pressione per far uscire la punta. La BOM di tale biro, che ha in comune con la BIC standard l'assieme cartuccia, è mostrato di seguito:

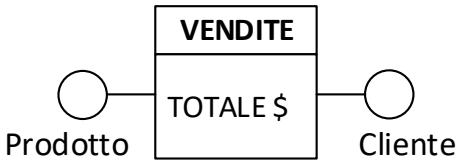
- Penna Tipo BIC Avanzata
  - Assieme Cartuccia
    - Cilindro di Plastica
    - Cartuccia
      - Cannuccia vuota
      - Inchiostro
      - Punta a sfera
  - Molla
  - Azionamento
    - Ghiera
    - Nottolino

Domanda #4 – 11 punti

Si considerino le quattro tabelle seguenti, relative al processo di acquisto di un cliente.



Supponendo che tali tabelle alimentino un Data Warehouse di tipo MOLAP, sintetizzato dal seguente diagramma DFM, si chiede di:



Descrivere la struttura della matrice multidimensionale che implementa tale modello (2 punti)

Creare 3 query atte, rispettivamente, a restituire le intestazioni di riga e di colonna della suddetta matrice e a generare la vista di partenza (con dati già opportunamente aggregati) necessaria alla creazione del DataWarehouse (3 punti)

Scrivere una procedura VBA atta a prelevare i dati dalle tabelle del DB relazionale e ad alimentare la matrice multidimensionale del Data Warehouse. (6 punti)

Suggerimenti:

Si crei una "*mappatura*" (connessione) tra i valori delle coordinate (etichette di righe e di colonne) e la loro posizione nella matrice. Un modo possibile, che non fa uso di dizionari o di altre strutture di dati, è mostrato di seguito, in cui i valori 2007, 2008, 2009 di un'ipotetica dimensione "anno" vengono usati come intestazione di colonna:

```
Dim Anno As Collection
Set Anno As New Collection
Anno.Add 2007, 1
Anno.Add 2008, 2
Anno.Add 2009, 3
        ….
Matrice(1, Anno(2008)) = 100 'Valore assegnato all'elemento (1, 2)
```

Si usino le query precedenti e opportuni recordset per popolare la matrice di dati, usando correttamente le mappature prima ottenute.

SOLUZIONE

Innanzitutto, si tratta di capire com'è fatta la matrice **M** atta a rappresentare il fatto di vendita indicato dal DFM. Banalmente, come indicato dal DFM, si tratterà di una matrice bidimensionale con dimensioni corrispondenti ai Clienti ed ai Prodotti. Inoltre il generico elemento ***M[c,p]*** della matrice corrisponderà agli acquisti totali (valorizzate in euro) del prodotto *p*, fatti dal cliente *c*. Si noti che abbiamo assunto implicitamente di mettere i clienti in riga e i prodotti in colonna, ma l'utilizzo della matrice trasposta sarebbe stato altrettanto valido.

Detto ciò si tratta di dimensionare la matrice ***M*** e di etichettarne righe e colonne, ossia di assegnare ad ogni riga uno specifico cliente e ad ogni colonna uno specifico prodotto. L'etichettatura servirà, appunto, a far sì che i valori della matrice possano essere prelevati per "chiave", oltre che per indice posizionale. Se ad esempio la prima riga corrispondesse al cliente Rossi e la prima colonna corrispondesse al prodotto Penna_BIC, allora l'etichettatura farà sì che le due seguenti notazioni siano del tutto equivalenti:

M(1,1) analoga a M("Rossi", "Penna_BIC")

Tornando al problema di fondo, sia per dimensionare che per etichettare la matrice, dobbiamo sapere quanti e quali clienti hanno effettuato un ordine e quanti[1] e quali prodotti sono stati acquistati. A tal fine sono necessarie queste due semplici query:

```
SELECT DISTINCT PRODOTTI.Nome
FROM PRODOTTI INNER JOIN DETTAGLI ON PRODOTTI.ID = DETTAGLI.ID_PR
```

```
SELECT DISTINCT CLIENTI.Cognome
FROM CLIENTI INNER JOIN ORDINI ON CLIENTI.ID = ORDINI.ID_Cl
```

La prima restituisce la lista dei prodotti acquistati (le etichette di colonna), la seconda la lista dei clienti che hanno effettuato almeno un ordine. Si noti che non potevamo opere direttamente sulle anagrafiche (tabelle clienti e prodotti), perché così facendo avremmo potuto trovare anche clienti che non hanno fatto ordini e/o prodotti non ordinati. A tal fine è necessario usare query di join, rispettivamente su Dettagli_Ordini e su Ordini[2].

Aprendo due recordset su tali query siamo in grado di dimensionare la matrice che avrà tante righe quanti sono i record della seconda query e tante colonne quante sono quelle della prima query. Inoltre, mediante tali recordset siamo in grado di predisporre le due collection, che ad esempio potremmo chiamare "Colonne" e "Righe", che definiranno la mappatura tra codifica posizionale e codifica per chiave. Come suggerito, supponendo di aver aperto il recordset Rcs_Row sulla seconda query possiamo procedere così:

```
Set Righe As New Collection
i = 1
Do While Not Rcs_Righe.EOF
        Righe.Add Rcs.Items(0), i
i = i + 1
Rcs_Righe.MoveNext
```

[1] Qui con "quanti" non intendiamo il valore cumulato, bensì il numero di differenti prodotti (a catalogo) che sono stati acquistati

[2] Il perché è lasciato come semplice esercizio al lettore

```
Loop
```

In questo modo al primo cliente viene associato il valore 1 (ossia la prima riga) al secondo il valore 2, ecc.

Si tratta infine di scrivere la query che crea la vista con tutti i dati necessari a compilare la matrice. Tale query dovrà sommare i valori delle vendite aggregando per cliente e per prodotto, si ha allora:

```
SELECT CLIENTI.Cognome, PRODOTTI.Nome, _
_ SUM(PRODOTTI.Prezzo*DETTAGLI.Quantità*(1-DETTAGLI.Sconto)) AS Totale
FROM PRODOTTI INNER JOIN ((CLIENTI INNER JOIN ORDINI ON CLIENTI.ID = ORDINI.ID_Cl) _
_ INNER JOIN DETTAGLI ON ORDINI.ID = DETTAGLI.ID_OR) ON PRODOTTI.ID = DETTAGLI.ID_PR
GROUP BY CLIENTI.Cognome, PRODOTTI.Nome
ORDER BY CLIENTI.Cognome
```

Si ottiene in tal modo il valore totale degli acquisti di ciascun prodotto per ciascun cliente. In pratica, quindi, il terzo campo di tale query (quello etichettato con Totale) contiene i valori che dovranno essere messi nella matrice ***M***. Ad esempio supponendo che Rcs_V sia una recordset aperto su tale query, basterà scrivere:

```
Do While Not Rcs_W.EOF
r = Righe(Rcs_W.Fields(0))
c = Colonne(Rcs_W.Fields(1))
M(r, c) = Round(Rcs_W.Fields(2), 2) 'Arrotondamento alla seconda cifra decimale
Rcs.MoveNext
Loop
```

Si noti che Rcs_W.Fields(0) restituisce il nome del cliente del record corrente della Vista. Per tanto, Righe(Rcs_W.Fields(0)) restituisce il valore numerico della riga della matrice ***M*** a cui corrisponde il nome del cliente trovato.

Una volta compilata la matrice, i suoi valori possono essere ricercati per chiave. Ad esempio, se volessimo cercare il valore delle vendite (acquisti) del cliente Rossi per il prodotto Biro_BIC potremmo scrivere:

```
Val = M(Righe("Rossi"),Colonne("Biro_BIC"))
```

L'intero codice è disponibile nel file in allegato. Si noti che il codice è implementato sia in maniera "procedurale", come richiesto nel compito, sia utilizzando un'opportuna classe denominata MOLAP2. Si rimanda a tale codice per i dovuti approfondimenti