THE BCS PROFESSIONAL EXAMINATIONS
Diploma

April 2006

EXAMINERS' REPORT

Database Systems

General Comments
For some reason some students still had difficulty in indicating on the front sheet which questions they had answered. Perhaps student attention needs to be drawn to this and a sentence is required on the front sheet which states 'Please circle the questions you have answered'. The majority of students answered the required number of questions, only a small group answered 2-3 questions. Students must take care on producing legible answers with correct spelling and/or grammar.

Question 1
1. Security and integrity are two database terms which are commonly used together, yet each has a different meaning.

a) Distinguish between security and integrity by giving a BRIEF definition of each. **(6 marks)**

b) Explain what is meant by entity integrity and referential integrity **(4 marks)**

c) The insurance policies have attributes policyNo, startDate, premium, renewalDate, policyType. Policy holders have attributes holderNo, holderName, holderAddress and holderTelno. Write SQL statements to create table definitions for the Insurance Policy table and the Policy Holder table. Make sure you use appropriate domains/data types for each attribute and also define suitable primary keys for each relation. A policy holder can have more than one Insurance Policy so you must define a suitable foreign key in the appropriate relation. **(6 marks)**

d) Give a detailed explanation of the database features that can be used to achieve security and integrity. Give examples using a DBMS you are familiar with to support your answer. **(9 marks)**

Answer Pointers
a) security is to do with preventing unauthorised access or malicious actions on the data whereas integrity is concerned with the accuracy and consistency of data in the DBMS.

b) Entity Integrity – each entity is uniquely identifiable using a primary key. The key must be unique and not null.
Referential integrity is concerned with maintaining consistency across related tables. In other words a table cannot have a foreign key value in a column where the value does not exist in the main table. This would be nonsensical.

```
c) CREATE TABLE  Insurance_Policy ( policyNo NUMBER PRIMARY
                                                        KEY,
                                   startDate DATE, NOT NULL
                                   premium DECIMAL(5,2) NOT NULL,
                                   renewalDate DATE NOT NULL,
                                   policyType VARCHAR(15) NOT NULL,
                                   holderNo number,
                             foreign key (holderNo) references
   policy_holder(holderNo));
```

```
CREATE TABLE Policy_Holder ( holderNo number primary key,
                             holderName varchar(25), NOT NULL
                             holderAddress varchar(50),
                             holderTelno varchar(20));
```

The foreign key should be defined in the Insurance_policy table to avoid repeating groups. Appropriate use must be made of NOT NULLS.

d) Marks spread equally across following points:
- user authentication using user id and password, password aging and complexity checks
- encryption to enhance security
- use of views to restrict data available to users
- use of grant/revoke to control access to relations and privileges
- use of roles
- use of entity integrity rule i.e. primary key definition whereby key column is not null and unique
- use of referential integrity i.e. foreign key clause to control consistency across related tables
- extra marks for good quality examples to support the above

Examiner's Comments

This question was less popular than question 2. Most students answered part a) and part b) without any problems. With part c), many students either forgot to put the foreign key clause in or put it in the wrong relation and lost marks as a result. Part d) was answered quite well. Detailed comprehensive answers gained higher marks.

Question 2

2. *a)* Describe the ANSI-SPARC three-level architecture under the following headings:
i) The external level, the conceptual level, and the internal level.
ii) The external/conceptual mapping and the conceptual/internal mapping.

Illustrate your answer with examples. What would be the effect if a DBMS only supported the conceptual and internal levels but not the external level? **(9 marks)**

b) Describe how the three-level architecture provides both logical and physical data independence. Illustrate your answer with an example. **(8 marks)**

c) A database management system uses a data dictionary for holding meta-data.

i) Explain what is meant by meta-data. There are many data dictionary views (e.g. USER_TABLES, USER_SYS_PRIVS etc) which a user can access to obtain metadata. Give an example using sql code which extracts/manipulates meta-data from one or more data dictionary views of your choice.

ii) Provide a detailed description of the role of a data dictionary and meta-data in a database management system. **(8 marks)**

Answer Pointers

ai) External level – user view of the data
Conceptual level – global view of the data
Internal level – physical view of the data

ii) the external level is derived from the conceptual level. You cannot have an attribute in an external schema which does not map onto a corresponding attribute in the conceptual level. In SQL the mapping is done when a view is created. We have to specify which table the view is based on and the order of the attributes in the view must match the order of the attributes in the base table when defining a view. This is how the mapping takes place. The mapping information is stored as meta-data.

Conceptual/Internal mapping – maps tables to indexes and files on the operating systems. Offset & byte information is given.

b) Logical data independence concerns external level and conceptual schema. It allows changes to be made to the conceptual schema without necessarily affecting the external schema. Physical data independence concerns conceptual schema and internal schema. The conceptual schema is immune to changes in the internal level. Definitions along these lines together with examples of each type of d. independence should be given.

c) i) meta data is "data about data" i.e. descriptive data.
Sample sql code: select table_name from user_tables;
This would retrieve all of the tables that the user has created and is the owner of.
select * from user_sys_privs where username like 'id01'; This would retrieve all of the system privileges that user 'id01' has been granted and also whether he/she has admin option for each of the privileges.

These are just two examples, students can choose any dictionary view they like and demonstrate an understanding of what that view is used for and how information can be retrieved from it.

ii) Data dictionary is a centralised repository of metadata. It is used for holding information on objects such as views, tables, indexes, clusters to show when they were created, who the owner is, column information, check constraints, etc. Security information is also stored in the data dictionary e.g. system privileges and object privileges. When a user creates a table, the dbms checks the data dictionary to make sure a table of that name does not already exist in the user schema. If it does an error message will be produced. So a data dictionary can be used for carrying out data validation.

When a user enters a query, the dbms will check the dictionary to make sure the query table exists and the query columns are correct for that table. If a value is being entered for a column which has a check constraint on it, the dbms will check the dictionary for the constraint and make sure the value fulfils that constraint i.e. is within a range or within a set of allowable values. E.g. if there is a constraint for colour where the allowable values are 'red', 'green', 'blue' and a user tried to enter 'purple', this data entry would be rejected.

Examiner's Comments

It is safe to say that most students had a good understanding of the three-level architecture and were able to give reasonable definitions in answer to ai).

However with aii) students tended to describe data independence when the question actually required them to show an understanding of how definitions at the external level are mapped onto/related to the conceptual level and the conceptual level is mapped to the internal level. As a plus point though, most students gave a good explanation of the effect of a DBMS supporting just the conceptual and internal levels but not the external level.

Part b) was answered well in general. Unfortunately the answers for part c) were very brief and gave the impression that students did not have much knowledge regarding the views in the data dictionary. Consequently the answers to cii) were of a similar quality. Only a handful of students scored high marks for part c).

Question 3

3. In web-based auction sites users submit bids and compete with other users bidding for the same product. Bidding continues for a specified period of time before the highest bidder (who becomes the buyer) secures the product. When the seller receives payment he/she posts the product to the buyer. A transaction is completed when the buyer receives the product. Since the seller is anonymous no correspondence from the buyer to the seller occurs, therefore a large amount of trust occurs. The integrity and honesty of sellers is recorded and this is made known to potential bidders.

a) Describe the requirements of a DBMS and database server needed to support the application outlined above. **(8 marks)**

b) Explain the interaction between a database server and a web server in order to present data stored in a database on a web browser. Illustrate your answer with references to application data and program code applicable to a web-based auction site. **(10 marks)**

c) Discuss the trade-offs of implementing the program logic and business rules on:
i) The application/web server;
ii) The database server. **(7 marks)**

Answer Pointers

(a) Marks allocated for issues pertinent to the application described:

- Protection/integrity of users: Access control and security
- Monitoring abuse of rules in the application
- Provide membership roles and establish rules/protocols
- Resource switching and connection time out
- Reliable with adequate backup control
- Extra marks for good examples

[8 marks]

(b) Marks spread equally for:

- Connection strings and Data Source Names
- The concept of ODBC and object level drivers
- How ownership of database equates to connection properties
- Grants and permission set for anonymous web user (e.g. ASP.NET)
- OPEN/CLOSE connection to named data source
- ASP.NET users may mention web config and other options
- Mention the use of cursors or equivalent data streaming measures

[10 marks]

Marks for emphasising choices and dilemmas:

- Transaction integrity (e.g. DTC – distributed transaction coordinator) being built into DBMS.
- Portability of code (e.g. poorer in DBMS because of lack of standards)
- SQL model better for closeness in mirroring I/O actions
- Performance DBMS better due to direct access to query optimiser and server processes
- Power and generality of OO programming frameworks

[7 marks]

Examiner's Comments

Most of the students who selected this question have answered it well especially sub question (b). However, few of students have explained the sub question (a) i.e. requirements of a DBMS and database server needed to support the e-commerce applications or (c) the trade-offs of implementing the program logic and business rules. The examiner suggests that candidates should 'read around' topics and also go through real example to appreciate the need for such three tiers architecture for e-commerce database applications.

Question 4

4. Consider the following database scenario:

“A part manufacturing company has one engineering department in Manchester and two manufacturing plants, one in London and one in Hong Kong. Each part type is produced at only one manufacturing plant. Currently, the company has one database located in the Engineering department in Manchester. Applications at the manufacturing plants access this database via a communication network for whatever data they need.

One of the relations in this centralised database system is the PART relation, where data about the manufactured parts are kept. The attributes of the PART relation are: the part’s number (Part#), part’s name (Name), part’s manufacturing cost (Cost), the part’s drawing number that specifies its design (Drawing#), the name of the plant where the part is manufactured (Plant), and the quantity manufactured up to now (Qty).

An instance of the **PART** relation is the following:

PART

Part#	**Name**	**Cost**	**Drawing#**	**Plant**	**Qty**
p2	Widget	200	123-7	London	500
p7	Gizmo	600	501-9	Hong Kong	1000
p3	Thing	100	238-2	Hong Kong	2000
p1	Gadget	1000	310-0	Hong Kong	40
p8	Acme	150	400-6	London	3000

The company has decided to move to a distributed database system where each of the sites has its own database.”

i) Propose a fragmentation design of the PART relation that reflects the distribution of the company’s sites and their functionality. **(8 marks)**

ii) Write an SQL statement for each of the fragments obtained in your design. **(9 marks)**

iii) Justify your proposed fragmentation design for the PART relation. **(8 marks)**

Answer Pointers

The student could either use relational a;gebra notation such as Π, σ etc.…or English words Project, select, Join, …

(a) What are all titles published by *Pitman*?

Project(Select BOOK (PName='Pitman') (Title)
OR
Π_{title} (σ $_{\text{PName='Pitman'}}$ BOOK)

(b) What are the specialisms of all authors publishing a book with *MIT Press*?

Project (Join .(Πt(Select BOOK($\sigma_{\text{PName='MIT}}$Press')(AName), AUTHOR:[AName=AName]) (Specialism)
OR
$\Pi_{\text{Specialism}}$. Π_{AName} ($\sigma_{\text{PName='MIT Press'}}$ BOOK) $\bowtie$ $_{\text{BOOK.}}$ $\sigma_{\text{AName=AUTHOR.AName}}$ ($\Pi_{\text{AName, Specialism}}$ AUTHOR))

(c) What is the location of the publisher of the book '*A guide to DB2*'?

Π (Join(Π (σ $_{\text{(Title= 'A Guide to DB2')}}$ BOOK) (PName)), PUBLISHER[PName=PName] (Location)

(d) Get the names of all publishers publishing a book by *Smith* and a book by *Jones*?

Divide (Π BOOK (ANname, PName), D:[AName | AName] where D(AName) is a relation containing two tuples-values Smith and Jones.

(e) What are the addresses of all authors publishing a book with all the publisher located in *Paris*?

Project (Join (Select (Join (Project BOOK (AName, PName)), PUBLISHER : [PName=PName]) (Location= 'Paris')), AUTHOR: [AName=AName]) (AName, Address)

Examiner's Comments

Most of the students have selected this question have not answered all the sub questions. It is apparent that students have been exposed to topic of distributed database concepts but did not practice enough to grasp the concepts. The examiner suggests that candidates should practice more distributed database applications to be able to design database applications which have advanced requirements which are not met by central database systems.

Question 5

Refer to Appendix A for this question

5. *a)* Construct an ER diagram containing Entity Types, Relationship Types and Degrees to give a high level model of the SWIFT information system described in Appendix A. Your model should have no more than 10 Entity Types and you should state the diagram notation you have used. **(10 marks**)

b) Extend your ER diagram by assigning the columns listed in the tables in Fig A1 and Fig A3 to the appropriate Entity/Relationship Types. Use your ER diagram to explain the interdependencies that exist between the data contained in Fig A1 and Fig A3. **(7 marks)**

c) Explain with the aid of examples obtained from Appendix A the concept of ***Domain Integrity*** in a data model and explain how ***Domain Integrity*** is translated into SQL code. **(8 marks)**

Important: STATE any assumptions made in your modelling**.**

Answer Pointers

First of all when starting an ER model it is important to identify a central entity type that will support the main transactions. If this was an order processing application then this would be an entity type called Orders. In the SWIFT discourse this would be an entity type called **Tickets**. Having established a starting point other types of entities are added that model data that **Tickets** needs to reference. These being Customers ; Seats; Flights ; Airlines and Prices. Do not include derived data – the high level data model does not elaborate any detail about attributes assignment. Then the main relationship types need to be added so that the transactions associated with pricing (derived) and offers are supported

Please note no marks were allocated for participation constraints and attribute assignment a skeleton ERD only accepted. Here are the suggested relationship types and degrees:

RT1: (Tickets → Customers) - a Ticket is purchased by one customer but a customer may purchase one or many tickets.
RT2: (Tickets → Seats) – a Ticket identifies a unique seat but that seat can be purchased many times for different flights.
RT3: (Tickets → Flight) – a Ticket is purchased for a particular flight but that flight has many tickets (allocated to seats).
The entity type Tickets could be thought of as a relationship between 3 entity types (Customers,Seats,Flights) forming a 3 –way or tripartite relationship.

Following on with the remaining Relationship types

Airline → Flights an airline offers many flights and a Flight is associated with many airlines. many to many (M:N) Create a linking entity Airline/Flight.
Relationship Type (RT) = Journeys
Flights → Offers a 1 to many relationship (1:M) RT=Has
A flight has offers for a particular flight and that offer only applies to a articular flight.
Seats -→ Offers (1:M) RT – Has
Similarly some seats are subject to an offer price but that offer price is associated with one seat. This allows for some flexibility in ticket pricing but this requirement is assumed and not stated in the discourse.
Prices → Tickets (1:M) RT For. This allows a look up of price ranges and discounts. For example a Price may apply to many tickets but a ticket has one price on a particular date.

Part b)
The ER model could be translated into tables rather than re-draw ERD. The attributes are ms

Airline(AirlineID AirlineName)
Customer(CustID, CustomerName etc ….
Tickets(Ticket#,FlightCode,SeatNo,CustID,PriceCode, PurchDate…

The Tickets entity shows the relationship between Flights, Customers and Seats by posting identifiers in the Ticket entity and adding a date for uniqueness. Therefore Ticket# becomes the candidate key.

Prices(PriceCode#,Price,DateValid
Offers(Offer#,AirlineID,Flight,NoTickets
# = PK

Other dependencies should be self explanatory as the hard work has been already done.

Part c)
Domain integrity is where a column in different tables share the same properties of data type, check constraint and other integrity constraints. Therefore dates are good candidates to be domains with appropriate integrity checks such as date within ranges of a flight being offered and the date of travel.

SQL has different methods of applying integrity depending on vendor support. For example there is an explicit CREATE DOMAIN or there is an implicit CREATE TYPE (abstract data type). So CREATE DOMAIN genericDates as SmalldateTime, check date is NOT .GT. getdate(‘today’)

Examiner’s Comments

Parts a) b)
Many candidates mixed up their answers to parts a and b in one single diagram. This is poor examination practice because if one part of the answer is wrong then the examiner cannot see the development of the answer. Many candidates lost marks because of this.
Another weakness was a failure to understand the simplification of ER models when 3-way relationships are used. If a 3-way relationship is not used there is a tendency to create a connection trap. The text book by ER Howe (data analysis for database design) illustrates the good practice mentioned above.

Parts c)
Generally satisfied with common expressions of check constraints in the SQL standard but surprisingly few candidates realised that SQL supports CREATE DOMAIN – perhaps because of a limited exposure to a range of DBMS software and corresponding SQL syntax.

Question 6

6. Query 1 shown below relates to the tables given in **Appendix A** at the end of this paper.

Query 1:

```
SELECT Cname FROM tbl_customers
WHERE CustomerID  IN (
  SELECT d1.CustID
    FROM TBL_Tickets d1,TBL_Tickets d2
     WHERE d1.CustID = d2.CustID
     AND d1.FlightCode != d2.FlightCode
     AND d2.PurchDate != Getdate(today()))
```

a) Explain in words the purpose of Query 1 and work out, using the tables in Appendix A and work out what result is returned. **Please show your working.** **(8 marks)**

b) Explain what is meant by a JOIN operator and show how Query 1 could be re-expressed using ***explicit*** JOIN operators. **(7 marks)**

c) Describe the different types of JOIN operators that SQL supports and explain with the aid of examples how each type of JOIN is used in practice. **(10 marks)**

Answer Pointers

Part a)

Query 1 uses a subquery construct thus avoiding the use of an explicit INNER JOIN. In some cases a subquery is used where a query is difficult to formulate using explicit JOINS operators. Using the tables in Appendix A, the following result is returned:

In English find those customers (by name) who have purchased tickets on different flights to the one they are have already purchased on a different day to today.

Thus a resultset is shown bold to show the working out thus

TicketNo	**CustName**	*FlightCode*	*PurchDate*
1966	**984311**	*PD0045*	*12/Nov/05*
1967	**984311**	*JDYE_6*	*10/Nov/05*

The customerID 984311refers to Hutton

Some working out should be shown in the candidates answer book hence reflected in question

Marking scheme:

2 marks for detecting the subquery;

3 marks for expressing in English the query;

3 marks for result set + working out

Part b)

```
SELECT CName
FROM tbl_customer AS a INNER JOIN TBL_Tickets AS p
  ON a.Custid = p.Custid
AND tbl_Tickets as r INNER JOIN TBL_Tickets as q
ON r.custid = q.custid
AND r.FlightCode != q.FlightCode
    AND r.PurchDate != Getdate(today()))
```

Marking Scheme

3 marks for expressing the INNER JOIN operator correctly

3 marks for reflecting the SUBquery as an INNER JOIN

2 marks for general reformulation of other JOIN operators

Part c) Different types with examples are:
INNER JOIN (cf Query 1 used to restrict dangling tuples so maintain Referential Integrity)
CROSS JOIN, (cross product used in a data warehouse/hierachical data sets)
SELF JOIN (reference query1!! A table that is joined on itself)
OUTER JOIN (left and right means relaxing matching on one side of relationship allowing dangling tuples with null values joining on another column ie match Plus NULLS)
Explanation should centre on the set based nature of relations having matching columns that restrict the result set down depending on relational Join type. Mention should therefore be made to INNER JOIN Some SQL syntax references an explicit INNER JOIN operator, Good candidates should spot that the INNER AND SELF JOIN operators are implicit part of the query. The entire query *could be formulated* as a subquery thus avoiding explicit JOIN operators in the nested subquery, this requires a result that is passed down and it cannot involve itself repeatedly (a type of recursion). **E**

Examiner's Comments

Part a)
Surprisingly very few candidates could arrive at the correct answer. Many candidates did not show any working it is good practice to simply write down each expression as a series of intermediate results working from the bottom up (like an expression tree)

Part b)
Some good answers but usually from candidates who understood the query expression in the first part. To avoid ambiguity always use an explicit INNER to describe the JOIN used. INNER is a default in implementing the SQL code, but when writing down code in an examination do not assume the default applies – the examiner needs to know if its intended or not.

Part c)
Most candidates got the popular types of JOINS but many missed the self join which actually is expressed in the code fragment for query 1! Examples have to make an impact – meaning they clearly show the difference, so examples drawn from the same tables should achieve this.

Appendix A: SWIFT (Saving With Internet Flight Travel) for use in answering Questions 5 and 6.

SWIFT is an agency that sells discounted seats on flights on behalf of some low cost airlines. These discounted seats are called 'Offers' and are only available for purchase by registered customers over the internet. There are a number of low cost airlines that SWIFT is an agency for and sometimes more than one airline can offer seats with the same start and destination. Each seat offered for sale contains some information that is supplied by the airline operator:-

- The Ticket Number that identifies the seat on a particular flight.
- The start and destination of the city/airport for the journey undertaken.
- The date and time of the flight/journey and the check-in time.

Some information is derivable and depends on the number of confirmed customer bookings/sales of seats, this information is held on the SWIFT database:

- The baggage allowance. The baggage allowance for passengers is reduced if the aircraft is carrying cargo as well as passengers. .
- The price of the seat. The price starts at a low discounted price and increases as the bookings fill up. The earlier the seat is purchased then the cheaper the price that a customer pays. The current price is the price quoted for customers wishing to purchase seats. This may also be the final price, particularly if customers book a seat near the departure date of a flight.

It is the responsibility of SWIFT to determine the price of a seat, they have to be competitive, however they must cover the operating costs of the flights. SWIFT have negotiated a contract whereby they get 12% of the profit that an airline makes each year from the passenger flights it is commissioned to sell seats.

Figure A1 (below) shows a sample ticket purchases and contains data that was captured today (assume the date is 3rd December 2005). **Figure A2** shows the customer details.

Figure A1: TBL_Tickets

TicketNo	CustID	FlightCode	Quantity	PurchDate	Current/Final Price	Price Paid	FlightDate/Time
1962	343371	BG_8971	1	12/Nov/05	231.99	219.56	01/Dec/05 12:23
1963	343371	GTX_281	2	12/Nov/05	211.67	211.59	14/Nov/05 16:34
1964	343371	TL121_281	1	12/Nov/05	2040.72	833.59	04/Dec 05 00:56
1965	034933	GTX_281	15	12/Nov/05	299.69	209.59	16/Nov/05 04:22
1966	984311	PD0045	5	12/Nov/05	102.67	99.59	16/Nov/05 06:09
1967	984311	JDYE_6	10	10/Nov/05	251.99	222.37	30/Nov/05 05:55
1968	343371	TL121_281	1	02/Dec/05	2040.72	2040.72	04/Dec/05 00:56
1969	953534	GTX_281	1	03/Dec/05	21.59	NULL	NULL

Figure A2: TBL_Customers

CustomerID	Member	CustomerName	Address1	Address2
343371	Y	Andrews	123 Abel Ave	London
034933	Y	Ling Wing	6 Princes Street	Hong Kong
984311	N	Hutton	564 Holly Road	Manchester
953534	N	Rivers	80 Grange Way	Glasgow

Figure A3 shows the flights that SWIFT have advertised on their web site.

Figure A3 TBL_Flights

OfferNo	AirlineID	FlightCode	Operating Cost	Seating Capacity	Flight Time	FlightDate
1004	B&G	BG_8971	20119.56	102	12:23	01/Dec/05
1005	B&G	GTX_281	19012.59	261	16:34	14/Nov/05
1006	JH Price	TL121_2	2033.59	100	00:56	17/Nov/05
1007	B&G	GTX_281	19012.59	260	04:22	16/Nov/05
1006	B&G	PD0045	21201.59	261	06:09	17/Nov/05
1008	JH Price	JDYE_6	22008.37	90	05:55	30/Nov/05
1009	B&G	JDYE_6	22008.37	100	11:23	30/Nov/05