THE BCS PROFESSIONAL EXAMINATION
Diploma

October 2004

EXAMINERS' REPORT

Database Systems

Question 1

1. *a)* In your own words, briefly describe *why* a relational database design must be normalised prior to SQL implementation. You should include any issues which may arise from using un-normalised relations. (You **DO NOT** need to describe the detailed mechanics of normalisation but you **DO** need to comment upon the role of functional dependency). **(5 marks)**

b) Using your own simple examples, describe in detail the objectives (i.e.what is to be achieved) and the mechanics (i.e. how it is achieved) of each of the following normalisation stages. (You should annotate and explain your examples clearly):

i) First Normal Form (1NF)
ii) Second Normal Form (2NF)
iii) Third Normal Form (3NF)
iv) Boyce-Codd Normal Form (BCNF) **(20 marks)**

Answer Pointers

(a) Main issues are (marks spread evenly across such comments):

- What is functional dependency between attributes?
- Un-normalised relations could incur insert/update/delete anomalies
- Normalisation as a process of decomposition
- The need for loss-less decomposition
- The need for a common attribute between decomposed relations
- Similar sensible comments

(b) Five marks for each stage. Each stage to include:

- The stage objective (for example, in 2NF to identify partial dependencies etc)
- The stage mechanics - how the identified problem is resolved (for example, in 2NF the removal of the determinant and the determined attributes into a new relation with the determinant attribute common to both (new) relations) = loss-less decomposition
- A simple annotated example

Examiners' Comments

Students were asked to give reasons why database should be normalised and what happens if not normalised. Most of the students gave detailed reasons for normalisation and also listed anomalies if not normalised.

On the question of describing in details the objectives and the mechanics of each of the first four-normalisation stages, some students gave good answers whilst others rambled, confused between the objectives and mechanics of the different normalisation stage. Reasonable attempts.

Question 2

2. Using your own simple SQL code examples, discuss and explain how SQL handles the following SELECT-related constructs. You must clearly annotate and explain each example used and comment upon any relevant issues.

a)	Set Operators	**(5 marks)**
b)	Joins	**(5 marks)**
c)	Logical Operators	**(5 marks)**
d)	Character Matching	**(5 marks)**
e)	Aggregate Functions	**(5 marks)**

Answer Pointers

Main issues for each category:

Set Operators

- UNION, DIFFERENCE (MINUS), INTERSECT etc
- Set operators work on sets of rows in the form of intermediate results sets
- The need for union compatibility between intermediate results sets
- Simple SQL example
- Possible Venn diagrams

Joins

- The join as a merging operation between two or more tables
- The different types of join (inner, outer, left, right, full, theta etc)
- The need to have a common linking column between two or more tables
- The need for those linking columns to share the same domain
- The fact that common linking columns do not need to have the same name
- The fact that some tables cannot be joined
- The role of the dot notation in avoiding ambiguity (table.column etc)
- The merging of the two common linked attributes in the final table
- Simple SQL example

Logical Operators

- AND, OR, NOT
- Use in WHERE clauses
- Use in constructing complex, multi-part queries
- Simple SQL example

Character Matching

- Need for exact and fuzzy matching in the WHERE clause
- Use of LIKE operator
- Use of wildcards (% and _)
- Comparison to = operator and exact matching
- Need for use of quotes with character strings (eg WHERE xxxx like ‘Smit%’)
- Consideration of case-sensitive searches (eg ‘SMITH’ ≠ ‘Smith’ etc)
- Simple SQL example

Aggregate Functions

- MIN, MAX, SUM, COUNT, AVG etc
- Concept of applying these functions to sets of values to produce a single result
- The fact that aggregate functions cannot be directly used in WHERE clauses (need sub-selects when comparing individual rows with an aggregate of rows)
- The problem of mixing aggregates with normal columns - for example, SELECT ColumnA, MAX(ColumnB) will not work (see next point)

- The role of aggregate functions in GROUP BY and HAVING constructs
- Simple SQL example

Examiner's Comments
Generally badly done – students either had no idea at all or confused set operations with the SET clause used with the UPDATE statement. A few students did give good conceptual answers (including Venn diagrams) but very few gave actual SQL code examples using UNION, INTERSECT etc

Joins **(5 marks)**

This was answered much better – most had a good SQL join example and many covered the different flavours of joins (INNER, OUTER etc).

Logical Operators **(5 marks)**

A few very good response but many students confused logical operators with relational operators (>, < , >= etc). For some reason, 'AND' and 'OR' were better covered than 'NOT'.

Character Matching **(5 marks)**

Generally good – nearly all students picked up on the LIKE operator and most covered the wildcards % and _ (although many did not differentiate between the two).
Some very good SQL examples provided.

Aggregate Functions **(5 marks)**

This was by far the best of the five sub-questions. Just about every student did well on this. Many good SQL code examples demonstrating AVG, MIN, MAX, COUNT etc and the better answers went on to cover GROUP BY.

Question 3
3. For EACH of the following database techniques explain how data security is maintained. Provide examples (in SQL) and diagrams (where applicable) to illustrate the implementation of these techniques**.**

a) database access control **(10 marks)**
b) database privileges **(10 marks)**
c) SQL *views* **(5 marks)**

Answer Pointers
(a) Marks spread equally over:
- Operating system versus database access levels
- Separate & discrete access versus cascaded access (database reads OS input)
- User IDs, passwords and DB connection strings (logging on to targeted DB)
- Password aging and complexity checks
- Password storage issues (encryption?) and the Data Dictionary
- Extra marks for good examples & diagrams

(b) Marks spread equally over:
- The concepts of grantor, grantee, object and object privilege
- The concept of privileges as a security feature
- Cascaded privileges
- System privileges versus object privileges

- Different privilege sets for different database objects (8 privileges for a table)
- Use of data dictionary to monitor issued privileges
- Relationship to roles
- Extra marks for good examples & diagrams

(c) Marks spread equally over:

- Views as a subset of base tables (subset of rows & columns)
- Views hide confidential data from un-vetted/inappropriate staff
- Views can have different privilege sets than constituent base tables
- Updateable versus non-updatable views
- Views storing derived/computed/modified data from base tables
- Extra marks for good examples & diagrams

Examiner's Comments
(a) Database Access Control – In the round, this was badly done. Relatively few students concentrated on login and password issues and instead get bogged down in the details of database privileges (the next sub-question). Some students got side-tracked into data locking techniques when handling concurrent access. This question was about authentication, not transaction processing.

(b) **Database Privileges** – Most students obviously had a good grasp of the core concepts and issues and most gave good GRANT & REVOKE SQL examples (although some gave incorrect syntax). Very few went on to cover roles or cascaded privileges. The different types & levels of user (from DBA down to end-user) was well explained, as were the different types of access (INSERT, UPDATE etc).

(c) Views – Mostly good – nearly all students gave a good CREATE VIEW example and explained the concept of data hiding by including sub-sets of base table columns.

Question 4

4. *a)* Provide practical examples of the threats and risks that a DBMS must deal with to preserve data accuracy and integrity. **(8 marks)**

b) Explain the techniques and mechanisms that a DBMS provides to preserve the accuracy and integrity of the data stored in a database system. Give examples using a DBMS product with which you are familiar. **(10 marks)**

c) Describe a range of design techniques that can be applied to identify potential threats on the data accuracy and integrity of data held in a database. **(7 marks)**

Answer Pointers
(a) Marks allocated for a wide range of issues:

- Concurrent access to the same data item
- Data input and validation according to prescribed data types/lengths
- Range and check constraints
- Entity and Referential integrity
- Access and restrictions
- Extra marks for good examples

(b) Marks spread equally over:

- The concepts of concurrency control and Locking access
- The concept of access roles and views

- Cascaded updates and deletes
- Schema definition
- Different techniques to implement input validation – TRIGGERS etc
- Use of data dictionary/catalog to dynamically capture constraints
- Extra marks for SQL/good examples from a DBMS eg Oracle/SQLServer

(c) Marks spread equally over:
- Transaction Analysis and CRUD matrix
- Business Rule analysis
- UML sequence/collaboration diagrams
- Extra marks for good examples & diagrams

Examiners' Comments

In general answers produced for this question was quite disappointing. It appears many candidates had simply not read the question carefully enough and immediately picked up a single topic area and wrote all they knew it. In fact the question covered more than one perspective – accuracy and integrity arising from data input and arising from lack of internal support to handle concurrent users they are different things. As a result candidates seemed to mix up their answers over all three parts to this question. In many cases candidates simply repeated themselves in their answers to part b and c. Very few candidates seemed to have working/practical experience of using integrity techniques on software and using the tools supported by an actual DBMS.

Question 5

5. In web-based auction sites such as Ebay, 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 were 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

(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 (eg 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

(c) Marks for emphasising choices and dilemmas:
- Transaction integrity (eg DTC – distributed transaction coordinator) being built into DBMS.
- Portability of code (eg 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

Examiners' Comments

On the question of describing the requirements of a DBMS and database server needed to support the E-bay web-based auction sites, only a small number of students have answered most of the requirements.

On the question of explain the interaction between a database server and a web server in order to present data stored in a database on a web browser, most of the students have described the three tiers architecture and technology needed to achieve communication between the tiers.

Also, most of the students have discussed very well the trade-offs of implementing the program logic and business rules on either the application/web server or the database server.

Question 6

6. *a)* State the advantages of *Entity-based* data modelling over *Relational-based* data modelling when designing data-centric applications. **(4 marks)**

b) Outline how an Entity-based data model can be translated into a Relational-based data model or schema. **(5 marks)**

c) Produce an Entity-based data model for the scenario in the Appendix on the next page. Your model must convey sufficient detail for translation to a Relational-based data model or schema – **DO NOT PERFORM THIS TRANSLATION**. You must state the notation you used in your model and state any assumptions you have made. **(16 marks)**

Answer Pointers

(a) The main advantages are better abstraction of the problem domain, closeness to real world objects and top-down approach better suited for large scale models (as opposed to bottom-up approach via normalisation).

(b) The mapping rules basically conform to 2NF that is entities become relations and relationships become relations when the relationship type is many to many constructing a relationship relation as a result (combined keys from the entities at either side of the relationship). When mapping a one to many relationship there are two options: Where participation on the many side is obligatory then two tables are the result, one holding a foreign key to a primary key in a base/entity type. Otherwise non-obligatory participation on the many side means three tables are required. For example a department has 1 or many employees results in two tables whereas a department has 0, 1 or many employees results in three tables (employees, departments and employee-departments).

(c) There is no prescribed ER model as candidates will interpret the scenario slightly differently. However the question was marked for accuracy and expressiveness using the description provided. Anything that contradicted the scenario did not gain any marks unless there was an explanation. Marks are allocated for identifying the main Entity Types :

Hotel, Date slot, Guest, Room (along with appropriate allocation of important attributes (eg status , dates, occupancy, room type) along with the more trivial ones (eg names/addresses etc).

The main relationship types (which can also become entity types if many to many relationships are eliminated) and degrees are apparent from the scenario:

Contains A hotel chain has many hotels each containing many rooms. Therefore a booking needs to specify the hotel and the room number.

Occupies: a room is occupied (has a confirmed booking) by one or many guests.

Reservation: guest can reserve one or many rooms at the same or different hotels but the date, room no. and hotel name must be unique). Similarly, a room may be reserved by one or none guests at any particular date. The model must reconcile block bookings spanning successive days, this is suggested to include a separate unit of booking for an arrival date and a leaving date rather than have unique rows for each day – the scenario indicates the way that a spreadsheet models this, in fact candidates should realise that this would be an inefficient way of modelling this in an Entity model but could be easily realised using appropriate code to manipulate the model.

Examiners' Comments

Parts a) and b) were generally well answered and it was pleasing to see a range of valid approaches to mapping ERDs to Relations rather than a stock answer. This means candidates understood part of a methodology they were actually using in practice. The only negative comment to part b) was the number of candidates who went to extraordinary lengths to describe the mapping process. The best thing that can be said about part c) was the gradual improvement that is being made each year at data modelling. But there is still some concern that candidates do not read the scenario carefully enough resulting in vague and inaccurate modelling assumptions. The bulk of the answers seemed to concentrate on the peripheral supporting entity types rather than the central transaction entity types and those that support the business rules (e.g. Bookings, Reservations). Existing confirmed bookings and residency were not mentioned in the scenario but candidates should have catered for this by including a status of a booking.