BCS THE CHARTERED INSTITUTE FOR IT

BCS HIGHER EDUCATION QUALIFICATIONS
BCS Level 6 Professional Graduate Diploma in IT

ADVANCED DATABASE MANAGEMENT SYSTEMS

Wednesday 30th March 2016 - Afternoon
Answer **any** THREE questions out of FIVE. All questions carry equal marks.
Time: THREE hours

Answer any Section A questions you attempt in Answer Book A
Answer any Section B questions you attempt in Answer Book B

The marks given in brackets are **indicative** of the weight given to each part of the question.

Calculators are **NOT** allowed in this examination.

Section A

Answer Section A questions in Answer Book A

A1

The hosting of database services on the 'Cloud' is representative of a Service Oriented Architecture (SOA). The SOA Company that hosts a client's database distributes their database physically and geographically, replicating their data across multiple database servers.

a) Outline the benefits of the distributed database architecture described above as opposed to a database that is centralised and not hosted. Illustrate your answer with a scenario of your choice that describes a database application that would benefit from this approach.

(7 marks)

ANSWER POINTER

Companies that operate worldwide/globally would benefit as the database services can be located close to their operational bases in these countries/cities. Reliability and high availability (continuous operation 24x7) are the key benefits. Many issues arise such as security, hosting cost model and autonomy – no reliance on a central site anymore, but issues have to be in place to secure these benefits. Flexibility of SOA in providing web based API hardware and maybe software neutral allowing greater choice of platform to deploy client applications.

4 marks

Scenario - a few sentences specifying what the database application does such as big data or global logistics for example is sufficient so that it would concur with the benefits above.

3 Marks

b) Describe the concept of **data replication** as a particular technique for distributing data and explain using your application as a source of examples how this is achieved.
(Hint: use diagrams and examples to illustrate how a distributed database is configured and accessed)

(9 marks)

ANSWER POINTER

Specifically address data replication and distinguish replication from 'true distributed database' concepts found in text books. For example, Replication technologies are concerned with copying and distributing data and database objects from one database to another and then synchronizing between these databases to maintain consistency. The issue is the latency of synchronisation which depends on the integrity of the database following updates and cascading these updates across sites. Using replication, you can distribute data to different locations and to remote or mobile users over local and wide area networks, dial-up connections, wireless connections, and the Internet.
Transactional replication is typically used in server-to-server scenarios that require high throughput, including: improving scalability and availability; data warehousing and reporting; integrating data from multiple sites; integrating heterogeneous data; and offloading batch processing. Replication is well suited to mobile applications or distributed server applications that have possible data conflicts. Common scenarios include: exchanging data with mobile users; consumer point of sale (POS) applications; and integration of data from multiple sites. it can also be used when complete refreshes of data are appropriate.

5 marks

Examples drawn from application should be informative and indicate from the perspective of the SOA company how data is replicated across sites.

4 marks

c) Given that the fundamental principle of a distributed system is that to a database user a distributed database system should behave exactly like a non-distributed system. Discuss the implications of the above statement on database integrity and the consistency of distributed queries and transactions in a distributed database that supports replication.

(9 marks)

ANSWER POINTER

Concurrency control involves locks and lock management that is no longer held centrally and there is an increased risk of global deadlock. Sites may need to interchange wait for graphs.
In practice Replication methods fail to achieve global lock management and 3PL techniques are not considered practical.

Recovery control if a transaction fails then rollback must consider the situation at other sites affected. Also take account of sites that are temporarily down or being updated when transaction was active. Replication of Transactions is costly and usually local database updates

its data and synchronises with main site meaning rollback is only local to the data that was changed

Update propagation if copies of data are kept to improve transaction performance of local sites then these copies must be updated.

Catalog or data dictionary should be managed globally yet changes might only affect local copies. Again an issue of managing changes to ensure consistency.

3 marks each for any of 3 issues above each for example

EXAMINER COMMENTS question A1

About a third of candidates attempted this question. There was a wide distribution of marks with just over half of candidates achieving a pass mark. Overall most candidates could discuss the general nature of distributed databases but very few candidates could gain the extra marks that were awarded if candidates applied their knowledge to SOA/Cloud computing. Therefore, prospective candidates are advised to look into the growing area of 'Big data' and the traditional distributed database techniques including replication that make this possible.

Section a) the evidence suggests that many candidates avoided any discussion and drawing examples from traditional database applications - many of which were unrepresentative of Cloud/SOA. However, there were some good answers covering mobile applications that warrant a type of lazy replication of database applications in which recording of data is local while a person is out of office/in the field.

Section b) had some good responses but the evidence suggests that the majority had limited knowledge of replication. The balance between theory and practice was more weighted to the former with few candidates developing their application (in section a) to a replication scenario. The mobile/field worker application was the type of application that reinforced answers.

Section c) Overall there was a sound appreciation of the problems associated with consistency and integrity (ACID properties) but many answers lacked the depth of analysis expected at this level.

A2

a) Draw a Class diagram using a stated notation, for example UML, derived from the following schema.

(8 marks)

```
Class Furniture (
String furniture_id ;
String furniture_model;
String furniture_maker;
Date   date_of_purchase;
String colour;
String setFurniture_id (string furniture_id);
```

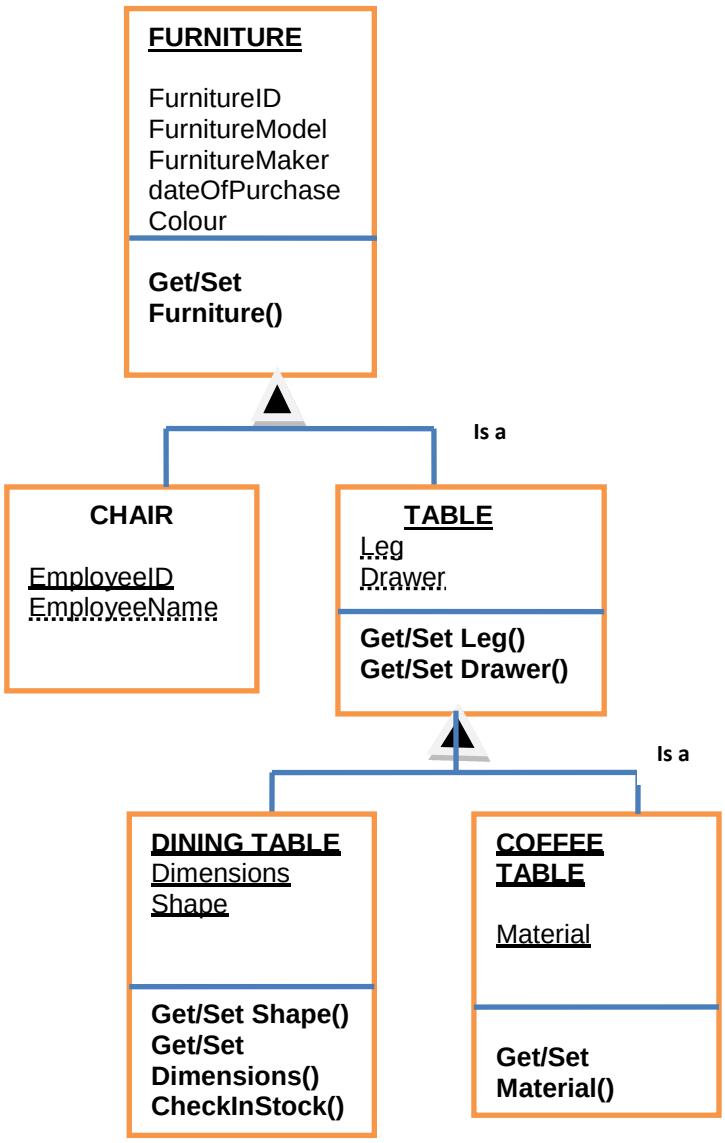
```
String getFurniture_id ();
)

Class table inherits from furniture (
Number leg;
Number drawer;
Number setLeg (number leg);
Number getLeg ();
Number getDrawer();
)

Class coffee_table inherits from table;
String material
String setMaterial (string material)
String getMaterial();
)

Class dining_table inherits from table (
String shape;
String dimensions;
String setShape (string shape);
String getShape();
String getDimension (string shape);
String setDimension(string dimension);
Boolean CheckInStock(string shape);
)

Class chair inherits from furniture (
String style;
String setStyle (string style);
String getStyle();
)
```

ANSWER POINTER

Marks allocated as follows:

3 marks for identifying classes, correctness of model, 3 marks for distinguishing methods from properties/attributes. 2 marks for inheritance relationships

Class UML data model



b) Explain each of the following Object Oriented data modelling concepts providing an example of each concept.

- Object Identity
- Object Relationships

- Object State

(8 marks)

ANSWER POINTER

An object is an entity that has state, behaviour, and identity. The structure and behaviour of similar objects are defined in their common class. For example, Furniture

Object identity — Objects are generally considered to have a unique identity; two objects which happen to have the same state at a given point in time are not considered to be identical.

We manipulate the State of an Object via its methods. And the methods define an object's "Behaviour", and its "States" are kept in Class variables.

Object relationships are between related instances that share a common property and may be part of relationships that involve association, aggregation, generalisation. Examples of these types of relationships are required in order to attain full marks.

c) There are a number of problems that arise in connection with the mapping of an object oriented system to a relational database and vice versa. These problems are referred collectively as *object-relational impedance.*
Explain the problems associated with object-relational impedance. Give particular regard to mapping the schema in part a) above to a Relational database.

(9 marks)

A rather philosophical question with the intention to restrict answers to the class model and its association with an equivalent Relational model

OR impedance concerns the conceptual and technical difficulties that are often encountered when a relational database management system (RDBMS) is being used by a program written in an object-oriented programming language or style, particularly when objects or class definitions are mapped in a straightforward way to database tables or relational schema.

Candidates should consider these problems related to the example OO model provided for example Class Furniture is modelled as a Table but that is far as it goes as normally Classes are not persistant and exist as programming constructs to reveal underlying behaviour

Classes vs Entity Types. OO encourages a tight association between verbs (actions) and the nouns (entities) that the operations operate on. The resulting tightly bound entity containing both nouns and the verbs is usually called a class, or in OO analysis, a concept. Relational designs generally do not assume there is anything natural or logical about such tight associations.

Object identity —Relations, have no inherent concept of this kind of identity. That said, it is a common practice to fabricate "identity" for records in a database through use of globally unique candidate keys; Relational systems in practice strive for and support "permanent" and inspectable identification techniques, whereas object identification techniques tend to be transient or situational.

Inheritance. Most relational databases do not support inheritance as OODM envisages by natural adoption of member instances/records. Hierarchical taxonomies and sub-typing are viewed as set-based classification in the RDM. OO advocates that inheritance/subtyping models need not be limited to single inheritance structures but non-tree OO solutions are seen as more difficult to formulate than set-based variation-on-a-theme management techniques preferred by relational.

Methods. In relational databases, attributes are accessed and altered through predefined relational operators, while OO allows each class to create its own state alteration interface and practices. In an object-oriented framework, the underlying properties of a given object are expected to be unexposed to any interface outside of the one implemented alongside the object. However, object-relational mapping necessarily exposes the underlying content of an object to interaction with an interface that the object implementation cannot specify. Hence, object-relational mapping violates the encapsulation of the object.

EXAMINER COMMENTS Question A2

An unpopular question with only around one fifth of candidates making an attempt. Those that did attempt scored well. There was a wide distribution of marks with over half of candidates achieving a pass mark.

Section a) the evidence suggests that candidates found this question difficult. The perceived difficulty in section a) was appreciating/modelling the concept of inheritance and the representation of methods and properties; in essence object orientated or extended ER (Entity Relationship) modelling. This topic is well covered in most database textbooks.

Section b) covered similar ground to section a) in that it addressed OO data modelling concepts and again it was a matter of knowledge.

Section c) produced generally poor answers though there were some exceptions.

In section b) many candidates had clearly not appreciated that the context of the question was security and therefore digressed from this topic in their answers. Consequently, marks were lost because candidates had either not read the question thoroughly or had only generic recall of integrity and availability.

In section c), knowledge of applying SQL and applying SQL specifically to security was patchy.

A3

(a) Consider the following three linked tables that contain information about students, the modules they are studying, and results of assessments for those modules:

```
students (studID, name, course)
modules (modNbr, title, credits)
results (studID*, modNbr*, date, grade)
```

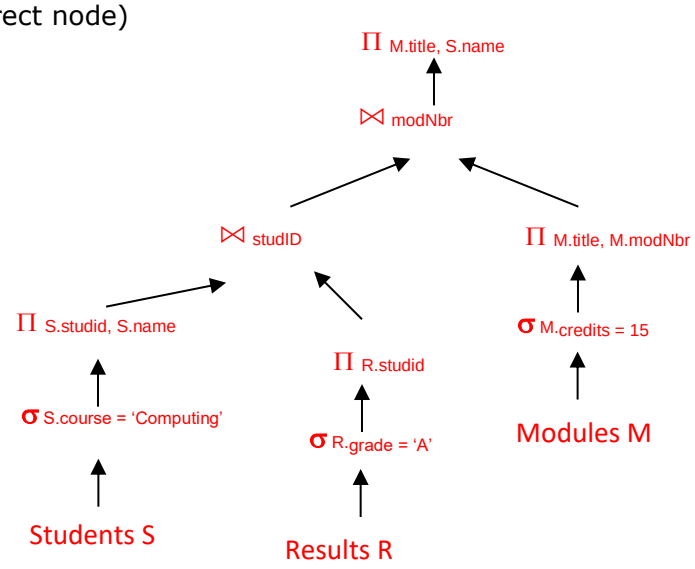
Consider the following query:

```
SELECT M.title, S.name
FROM students S, modules M, results R
WHERE S.studID = R.studID
AND M.modNbr = R.modNbr
AND S.course = 'Computing'
AND M.credits = 15
AND R.grade = 'A';
```

Draw a query tree that corresponds to the most efficient way of processing this query.

(12 marks)

(1 mark for each correct node)



(b) Securing a database aims to achieve the following objectives:

- Confidentiality
- Integrity
- Availability

(i) Briefly describe what is meant by each of the above objectives.

(3 marks)

(ii) For each objective, describe two appropriate security controls.

(3 marks)

ANSWER POINTER

(i) (1 mark each)

Confidentiality: information is not made available or disclosed to unauthorised users or processes.

Integrity: the property of maintaining and assuring the accuracy and consistency of data.

Availability: the property of ensuring data is accessible and usable upon demand by an authorised user.

(ii) (1 mark each) – Just two controls required per objective

Confidentiality: could be enforced, for example, using encryption of sensitive data, enforcing access control, the use of views, virtual private database,…

Integrity: could be enforced, for example, using entity, referential integrity and domain constraints during database design, and using access control, least privilege policy, separation of duties concept…

Availability: could be enforced, for example, using regular backups of the database, implementing hardware redundancy (RAID), providing appropriate bandwidth on the network,…

(c) Malicious users, such as hackers, can use some of the characteristics of the SQL language to their advantage.

(i) SQL is case-insensitive. Discuss why this fact makes it more difficult to defend against SQL injection attacks.

(2 marks)

(ii) SQL imposes a precedence rule on the use of logical operators: the AND operator has precedence over the OR operator. Using an example of your choice, illustrate how this precedence works.

(2 marks)

(iii) Explain how a hacker could take advantage of this precedence rule during an SQL injection attack.

(3 marks)

ANSWER POINTER

(i) (2 marks)

Protecting against common malicious SQL statements becomes infeasible if one decides to create a blacklist, for example, as there are so many ways in which a statement or key word could be written.

(ii) (2 marks)

For example:

```
SELECT *
FROM employees
WHERE salary > 20000 AND dept = 10 OR name like 'M%';
```

is equivalent to:

```
SELECT *
FROM employees
WHERE (salary > 20000 AND dept = 10) OR name like 'M%';
```

(iii) (3 marks).

Suppose a hacker wants to attack a web page that asks users for username and password. The underlying SQL could be something like:

```
SELECT * FROM users_table WHERE user = &userinput AND password =
&passinput;
```

A hacker can inject OR 1=1 making the statement behave as follows (given the precedence rule):

```
SELECT *
FROM user_tables
WHERE (user = &userinput AND password = &passinput) OR 1=1;
```

This evaluates to TRUE, which means that the hacker will be granted access and able to see the details of all users.

EXAMINER COMMENTS Question A3

Most candidates attempted this question with Section a) well established and well covered on previous papers. The average mark was relatively high and there was a reasonable distribution of marks with a large number of candidates achieving a pass mark.

Section a) was perceived to be quite straightforward, but performance fell away on sections b) and c). Therefore, the relatively high overall average mark can be attributed to section a) as it tended to compensate for the harder marks that may not have been attained in sections b) and c).

In part c), knowledge of applying SQL and applying SQL specifically to security was patchy either good or very poor. This indicates candidates are lacking practical skills on using SQL and rely too much on bookwork.

Section B

Answer Section B questions in Answer Book B

B4

(a) The design of data warehouses extends and enhances the underlying concepts from database design. For each of the following three approaches, using appropriate supporting examples of your own choosing, explain and discuss the essential concepts and design issues of each.

(i) Entity and Enhanced Entity Relationship Diagrams **(5 marks)**

(ii) Star Schemas **(5 marks)**

(iii) Snowflake Schemas **(5 marks)**

You should particularly address the roles of primary and foreign keys, normalized & de-normalized data. Good diagrams are essential.

ANSWER POINTERS

E/ERD: based on entities (logical or physical item of interest within a domain of discourse) and relationships (logical connections between entities), differentiating between entity (and relationship) types and entity (and relationship) instantiations. The concept of super-types and sub-types – as used on the EERD - should be explained – for example super-type STUDENT having sub-types UNDERGRADUATE AND POSTGRADUATE etc. Comments on attributes and the use of primary keys/identifiers. Good examples/diagrams expected.

EXAMINER COMMENTS

Mostly well answered with clear example ER diagrams. The sub-typing concept was less well covered.

Star Schema: a specialized example of an ER Model with the use of a *fact table* (with composite primary key) and a set of *dimension tables* (each with an atomic primary key) related to the fact table via foreign keys – thus producing a *star schema* (star join) model. The better students should then go on to discuss issues such as the fact table is much larger than the dimension tables, that the fact table works best when the 'facts' recorded are numeric (grades, prices, ages etc.) thus allowing aggregated computations to be run leading to summarized data, that dimension tables tend to hold more descriptive data (names, addresses, identifiers etc.), the use of de-normalized data to replicate attributes across multiple dimension tables (for example, storing address or contact data in several different dimension tables) thus avoiding additional joins and enhancing query performance. Good examples/diagrams expected.

EXAMINER COMMENTS

Almost universally well answered with clear diagrams illustrating fact and dimension tables and the roles of PK/FK. A good sub-question for those who attempted it.

Snowflake Schema: an extension of the Star Schema where dimensions can have their own dimensions – caused by normalizing the original dimension table data down into two or more child dimension tables, all linked to the 'parent' dimension table via the familiar PK/FK technique. So *star schemas* use de-normalized (repeated) data and *snowflake schemas* use normalized (minimized duplication) data. Good examples/diagrams expected.

EXAMINER COMMENTS

Again, a strongly answered sub-question with just about all students supplying a good-quality diagram.

(b) By their very nature, data warehouses get bigger over time. As the search space increases, the query performance decreases, and tuning techniques are required. For each of the two approaches listed below, using your own appropriate diagrams and examples, explain and discuss the essential concepts and techniques.

(i) Aggregation & Summary Data **(5 marks)**

(ii) Indexing & Optimization **(5 marks)**

ANSWER POINTERS

Aggregation: data warehouse queries are often seeking aggregated data, not the fine detail of individual rows, particularly aggregation via specific dimensions (month, product, region etc.) so the DBMS must support pre-computed summaries and aggregates to avoid run-time computation. Often used with *analytical functions* – many BI and DW applications want to use SQL ranking and aggregate functions, cumulative aggregates or maybe the CUBE and ROLLUP operators for OLAP analysis.

EXAMINER COMMENTS

Evidence suggests that there was a small minority of students giving strong, detailed responses covering the points flagged up in the marking scheme but with the bulk of answers being vague.

Indexing: The primary (but not only) method for query optimization. It can include:

- Bitmap indexes – via compression, the support for low cardinality data and more 'open' type of queries rather than the usual B-tree indexes used to search for individual identifiers.
- Advanced join methods - such as in-memory hash joins and partition-wise joins – very useful for the large data volumes involved in a DW.
- Advanced optimizers - that can use statistics and histograms - especially on skewed data distributions – such as Oracle's cost-based optimizer when working on star schema queries.

- Function-based indexes - that can pre-index complex calculations and other expressions often used in DW applications. Sampling functions that can gather mathematical data like averages without having to read every row.

EXAMINER COMMENTS

Those who did well covered bitmap indexes and cost-based optimizers but few got function-based indexes and even less on advanced join operations. However, for many students, this was a weak and vague question.

B5

Ensuring data integrity and consistency is of vital importance to a DBMS during the application of transactions to the database, particularly concurrent transactions. Using your own suitable examples and diagrams, explain and discuss the following transaction-related concepts. Five marks each.

(i) ACID Properties **(5 marks)**

(ii) COMMIT & TWO-PHASE COMMIT **(5 marks)**

(iii) ROLLBACK & CASCADED ROLLBACK **(5 marks)**

(iv) Locking – Optimistic & Pessimistic **(5 marks)**

(v) Checkpoints & Savepoints **(5 marks)**

ANSWER POINTERS

ACID: as the fundamental framework for transaction-processing, specifying atomicity, consistency, isolation and durability - each aspect must be named and explained in detail. Bonus marks for a clear worked example or well annotated diagram.

EXAMINER COMMENTS

Universally well done.

COMMIT/TWO-PHASE COMMIT: as a saving operation (of the complete transaction) – as opposed to ROLLBACK – see below. Students should then go on to describe the need for the two-phase commit in a distributed database environment and how it constitutes the 'voting' phase (the 'are you ready to commit' message) and the 'decision' phase (the 'commit now' message). Students should also cover the concepts of *global transactions*, *localized sub-transactions*, *transaction coordinator site* (the transaction initiator site), *participant sites*, the need to pass *messages* between sites, the use of *timeouts* to avoid unnecessary blocks or delays, the possibility of a participant site issuing a *local abort* – thus forcing a *global abort* to be issued and the need for unanimous *local commits* before a *global commit* is circulated – all ensuring the *data integrity* and *ACID rules* are satisfied. Bonus marks for a clear worked example or well annotated diagram.

EXAMINER COMMENTS

Again, strong responses on the COMMIT but less good on the 2PC.

ROLLBACK/CASCADED ROLLBACK: ROLLBACK to undo the whole transaction. Cascaded Rollback - when transaction T_1 fails and induces a rollback which in turn causes other transactions - which were dependent on the success of T_1 – to likewise fail and be rolled back. Bonus marks for a clear worked example or well annotated diagram.

EXAMINER COMMENTS
Another strong question for most students.

Locking: Optimistic locking – based on the assumption that inter-transaction conflict is rare so individual transactions are allowed to proceed unsynchronized and are only checked for conflicts at the end – just before commit. Useful in low data contention environments because it avoids the overhead of locking/waiting for unlocking but inefficient if data conflicts are common as transactions will have to be repeatedly restarted. Pessimistic locking – assumes a high degree of inter-transaction conflict and locks up all data resources ahead of access immediately – even if other transactions never try and access them. Saves re-running transactions in highly contentious environments but this 'belt and braces' approach can induce overhead in locking/releasing data resources that were never in conflict in the first place.

EXAMINER COMMENTS
The answers ranged from excellent to weak and very vague.

Checkpoints & Savepoints: Checkpoint – a point of synchronization between the database and the transaction log file (journal) that allows easy identification of which transactions need to be re-processed (redo/undo) in the event of database recovery being needed. SAVEPOINTS as transaction partitioning concepts whereby a large transaction can be sub-divided down into smaller units using embedded labels and how the DBMS can roll back to a named savepoint rather than undoing the whole transaction

EXAMINER COMMENTS
Savepoints universally well explained but checkpoints were not.