THE BCS PROFESSIONAL EXAMINATIONS
Diploma

April 2007

EXAMINERS' REPORT

Database Systems

General Comments

Some students (albeit a smaller number than last time) 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 fewer questions than the required amount. Students must take care on producing legible answers with correct spelling and/or grammar.

Question 1

Database Access control, database privileges and SQL views are three main techniques for maintaining database security.

a) Explain how EACH technique can be used to maintain data security. **(15 marks)**
b) Provide SQL examples to illustrate each of these techniques. **(10 marks)**

Answer Pointers

MODEL a) required students to explain how each technique (Database Access control, database privileges and sql views) can be used to maintain data security.

A definition for each of these techniques would be a useful starting point. Followed by a description of the purpose of each technique. Database Access control is the first security level which only allows authorised users who have a login account and the correct password to use the database and access the data. Some systems are set up so that a user only gets three attempts to login and if all three attempts are unsuccessful the user account is locked and he/she has to see the database administrator (DBA) to get their account unlocked.

If a user manages to log into a database then the second security measure is the use of privileges. The DBA issues users with appropriate database privileges so the users can carry out the work that they have been asked to do. Types of privileges are create, read, update and delete. Some users might only be given read access whilst others may be given the full set of privileges. This is something that is decided by the dba and others at management level. The other thing to mention is that users may be granted the right to issue the privileges they have got to other users. Again that is a decision made by the dba on whether he/she wants certain users to have that flexibility.

Finally SQL views offer a restricted view of data for users who do not need to see all of the data in the database. This feature is used mostly when the database contains data of a sensitive nature. Views do not actually store data, they are derived from the base tables. Some views can be updated others can't, it depends on how they have been set up. It is usually the dba and his data management team who decide on the policy of which views to create and which users can have access to these views.

MODEL b) required students to show their knowledge of sql and demonstrate how each technique is implemented using sql.
Database Access control – required students to give an example using the 'Create user … identified by …' command.
Database Privileges – students were required to provide examples which made use of 'grant' and 'revoke' statements. The 'with admin' option could also be discussed here to show fuller knowledge of this security technique
SQL views – students needed to provide an SQL 'Create view' statement demonstrating how it would show a restricted view of the data compared to the base table

Examiner's Guidance

It is important to answer all aspects of the question. Some students had a tendency to focus too much on one MODEL and neglect the other. This resulted in the student missing out on marks for the neglected MODELs. Most students were able to answer MODEL a) satisfactorily but MODEL 2) proved a bit more of a problem due to lack of sql knowledge.

Question 2

Relational theory and relational algebra are the foundation of modern relational databases. Describe and discuss the following: atomicity; entity integrity; referential integrity; union; project. **(25 marks)**

You should support your answer to *a)* and *b)* with your own simple examples and any appropriate diagrams.

Answer Pointers

Students should start off by defining each of the terms atomicity, entity integrity, referential integrity, union, project and then proceed to give a fuller description of each.
Atomicity – each row column intersection in a relation can only have a single value not lists of values.
Entity Integrity – each row must be uniquely identifiable. Hence the primary key must not be null and must not contain duplicate values.
Referential integrity – foreign key in one relation is primary key in another. This is used to manage relationships between relation correctly.
A relation can have more than one foreign key but only one primary key.
Union – combining data from two relations which are union-compatible ie they have the same number of columns which are based on the same domains. So a union could not be carried out on a relation which has two columns and another relation which has 4 columns because they do not meet the union-compatibility rule.
Project – extract certain columns from a relation rather than displaying all of the columns in the final result.
SQL examples and use of diagrams (where appropriate) were required to gain further marks for this question

Examiner's Guidance

The first problem was that students confused the use of the word atomicity with use of the same term in another context ie that of ACID properties of a transaction.
Some students wrote in length on ACID transaction properties and unfortunately did not gain marks because their answer to this MODEL of the question was irrelevant. Always read the question properly, think before you answer.
Some students got stuck on Union and project but on the whole this question had a higher pass rate than question one.

Question 3

The table below represents a sample report layout for a construction company that manages severalprojects. Each project has its own number (P-No), name (P-Name), and employees assigned to the project. Each employee has an employee number (E-No), name (E-Name), and a job classification (Job-Class).

The company charges its client by billing the hours spent on each contract. The charge per hour (Chrg-Hr) rate is dependent on the employee position or job classification (Job-Class). The total charges (Tot-Chrg) is the product of hours billed (Hrs-Billed) and charges per hour (Chrg-Hr).

A Sample Report Layout

P-No	P-Name	E-No	E-Name	Job-Class	Chrg-Hr	Hrs-Billed	Tot-Chrg
1	Harricane	101	John News	Elect. Eng.	65	13	845
		102	David Senior	Comm. Tech.	60	16	960
		104	Anne Ramoras	Comm. Tech.	60	19	1,140
2	Coast	101	John News	Elect. Eng.	65	15	975
		103	June Arbough	Biol. Eng.	55	17	935
3	Satellite	104	Anne Ramoras	Comm. Tech.	60	18	1,080
		102	David Senior	Comm. Tech.	60	14	1,920

A database designer was asked to develop a database from which the information contained in the above Sample Report could be generated. For this, he/she designed the **Project** table whose structure matches the above report formats. He/she omitted the total charge attribute because he/she thought that it could be calculated using charge per hour (Chg-Hr) and Hours billed (Hrs-Billed).

Project

P-No	P-Name	E-No	E-Name	Job-Class	Chrg-Hr	Hrs-Billed
1	Harricane	101	John News	Elect. Eng.	65	13
		102	David Senior	Comm. Tech.	60	16
		104	Anne Ramoras	Comm. Tech.	60	19
2	Coast	101	John News	Elect. Eng.	65	15
		103	June Arbough	Biol. Eng.	55	17
3	Satellite	104	Anne Ramoras	Comm. Tech.	60	18
		102	David Senior	Comm. Tech.	60	14

Answer the following *a)* and *b)* questions:

a) As the above **Project** table developed by the database designer is susceptible to update anomalies, provide ONE example of EITHER an insertion, deletion, OR update anomaly. **(10 marks)**

b) Using the functional dependency diagrams (fd1, fd2, etc…), describe and illustrate the process of normalisation from First Normal Form to Third Normal Form for the above **Project** table. In this process of Normalisation, we assume that the **Project** attributes P-No, E-No and Job-Class could be used to determine the values of (P-Name), (E-Name and Job-Class), and (Chrg-Hr) respectively. **(15 marks)**

Answer Pointers

Sub question a) required student to give an example of either inserting, deleting or updating one of the regards given in the following Project table:

Project

P-No	P-Name	E-No	E-Name	Job-Class	Chrg-Hr	Hrs-Billed
1	Harricane	101	John News	Elect. Eng	65	13
		102	David Senior	Comm. Tech.	60	16
		104	Ann Ramoras	Comm. Tech	60	19
2	Coast	1014	John News	Elect. Eng.	65	15
		103	June Arbrough	Biol. Eng.	55	17
3	Satellite	104	Anne Ramoras	Comm. Tech	60	18
		102	David Senior	Comm. Tech	60	14

An example of such operation could be updating the Job class of John News in the first record (which will be inconsistent with record 4 in the table), etc…

Sub question b) requires student to identify functional dependency fd1, fd2, etc…
Such as: fd1: P-No -> P-Name, fd2: E-No-> E-Name, Job-Class, Fd3: Job-Class-> Chrg-Hr and use these functional dependency to determine primary keys and use them to split (1st NF, 2ndNF and 3rd Form) and link the split table by foreign key (normalise the table). This will lead to the result tables:
Project (P-No, P-Name)
Employee (E-No, E-Name, Job-Class)
Project-Employee (P-No, E-No, Hrs-Billed)
Job Types (Job-Class, Chrg-Hr)

Examiners' Guidance Notes

Most of the students attempted this question. The majority of students have answered correctly sub question (a) on identifying anomalies. However, fewer students have answered correctly the normalisation process till the 3rd form of the sub question (b). The examiner suggests that candidates should undertake more exercises on Normalisation process from First normal form till at least the third normal form.

Question 4

Consider the following scenario:

"A high performance bicycle manufacturing company has two engineering departments one based in London and one in Manchester, and three manufacturing plants, one in Swindon, one in Hong Kong and one in Taipei. Each bike model is produced at only one manufacturing plant. To allow for sharing of data, the company has one database located in the London engineering department. 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 MODEL relation, where data about the manufactured bike models are kept. The attributes of this relation MODEL are: the model's code (Model#), model's name (Name), manufacturing cost (Cost), the drawing number that specifies its design (Drawing#), the name of the engineering department that engineered the model (Eng_Dept), the name of the plant where the model is manufactured (Plant), and the quantity manufactured up to now (Qty). An instance of the MODEL relation is the following:

MODEL						
Model#	**Name**	**Cost**	**Drawing#**	**Eng_Dept**	**Plant**	**Qty**
P2	Solo	£200.00	123-7	London	Taipei	50,000
P7	Sprinter	£600.00	501-9	Manchester	Hong Kong	1,000
P3	Interlude	£100.00	238-2	Manchester	Hong Kong	2,000
P1	Scarlet	£1,000.00	310-0	Manchester	Swindon	10
P8	Pelican	£150.00	400-6	London	Taipei	3,000

The company has decided to move to a distributed database system where each of the sites (engineering and manufacturing) has its own database."

a) Propose a fragmentation design of the MODEL relation that reflects the distribution of the company's sites and their functionality. **(7 marks)**

b) Justify your proposal. **(3 marks)**

c) For each fragment, give an SQL statement that defines it. Finally, give an SQL statement that reconstructs the original MODEL relation from its fragments. **(15 marks)**

Answer Pointers
Sub-question a) the student should identify the criteria of the table MODEL below he/she could use to split/fragment horizontally or vertically the table:

MODEL					
MODEL#	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

There might be different ways of distributing the MODEL table. One strategy of fragmenting the MODEL table is as follows:

First, the relation MODEL is firstly fragmented vertically. The first fragment (MODEL#, Name, Drawing#) is about MANCHESTER engineering data, whereas the second fragment (MODEL#, Cost, Plant, Qty) contains London and Hong Kong manufacturing data. Data about the Name and Cost attributes could be allocated in any site; however, "drawing" and "Qty" should be allocated to Manchester (where the design is done) and to both London and Hong Kong respectively (where the manufactured MODELs are stored).

Manchester-MODEL

MODEL#	Name	Drawing#
p2	Widget	123-7
p7	Gizmo	501-9
p3	Thing	238-2
p1	Gadget	310-0
p8	Acme	400-6

London-HongKong-MODEL

MODEL#	Cost	Plant	Qty
p2	200	London	500
p7	600	Hong Kong	1000
p3	100	Hong Kong	2000
p1	1000	Hong Kong	40
p8	150	London	3000

The London-HongKong-MODEL fragment is then fragmented horizontally into two groups of tuples (one for each engineering site) each of which is stored at the corresponding engineering site (i.e. London and Hong Kong). So London-HongKong fragment (MODEL#, Cost, Plant, Qty) is horizontally fragmented to two groups of tuples. The ones where Plant = "London" will be stored in London, those where Plant = "Hong Kong" will be stored in Hong Kong. The resulting fragmentation is as shown below:

London-MODEL

MODEL#	Cost	Plant	Qty
p2	200	London	500
p8	150	London	3000

HongKong-MODEL

MODEL#	Cost	Plant	Qty
p7	600	Hong Kong	1000
p3	100	Hong Kong	2000
p1	1000	Hong Kong	40

To eliminate the repetition of the Plant attributes "London" in London-MODEL fragment and "Hong Kong" in HogKong-MODEL fragment, the London-MODEL and HongKong-MODEL could be represented as shown below:

London-MODEL

MODEL#	Cost	Qty
p2	200	500
p8	150	3000

HongKong-MODEL

MODEL#	Cost	Qty
p7	600	1000
p3	100	2000
P1	1000	40

Sub question b) requires the student to justify his/her proposed fragmentation design for the MODEL relation.

The fragmentation could be a mixed fragmentation of the MODEL relation. First a vertical fragmentation based on the type of the plants i.e. engineering or manufacturing. As a result Manchester fragments will hold data (MODEL#, Name, Drawing#) necessary to the engineering deMODELments while London and Hong Kong fragments will hold information (MODEL#, Cost, Qty) on the MODELs built at the local plant. In the vertical fragmentation the attribute MODEL# is included in both the fragments so that the original relation can be

reconstructed. A further horizontal fragmentation is performed to allow information related to each plant to be held in a local database i.e. London and Hong Kong databases.

Sub question c) is to write an SQL statement for each of the fragments obtained in above (sub question a) design

SQL statement for Manchester-MODEL fragment:
Select MODEL, Name, drawing
From MODEL

SQL statement for London MODEL fragment:
Select MODEL, Cost, Qty
From MODEL
Where Plant=”London”

SQL statement for Hong Kong MODEL fragment
Select MODEL, Cost, Qty
From MODEL
Where Plant=”Hong Kong”

Finally the reconstruction of the MODEL table will be as follows:

(London-MODEL U Hong Kong MODEL) Join (Manchester-MODEL)

Examiners’ Guidance Notes

Few students have selected this question. It is apparent that students have not been well exposed to the topic of distributed database concepts and did not practice enough to grasp distributed database concepts. The examiner suggests that candidates should practice more on distributed database applications to be able to design database applications with advanced requirements which are not met by central database systems.

Question 5

Refer to the tables listed in **Appendix A** at the end of this paper.

a) Write SQL code that will create the Tables **Copy** and **OnLoan**. **(5 marks)**

b) Explain the function of the ALTER TABLE statement in SQL and explain how it would be used to add referential integrity between Tables **OnLoan** and **Copy**. **(6 marks)**

c) List the **dvdName** and **dvdGenre** of DVD's with dvdGenre = Animation that were ranked amongst the top 5 rentals last week and are still ranked in the top 5 for this week. **(8 marks)**

d) The Table **TopTenRentalsThisWeek** could be derived from existing data rather than be persisted as aTable. Explain how you would achieve this in SQL. **(6 marks)**

Answer Pointers
Reference was made to 5 tables listed in the Appendix. In preparation for writing SQL code candidates should draw instance connections between related columns in tables. For example dvdID 20467 in Copy.dvdID appears in TopTenRentalsdvdID (row 1) and in DVD.dvdID (row 3). This establishes the permissible JOINS and helps in understanding the SQL code that is required. Candidates should annotate the above as connecting lines on the Tables.
MODEL a) Code solution for creating the Tables **Copy** and **OnLoan**.

```
CREATE TABLE Copy(
copyID nchar(5) NOT NULL,
dvdID nchar(5) NOT NULL,
status nvarchar(50) NOT NULL);

CREATE TABLE OnLoan(
copyID nchar(5) NOT NULL,
CustID integer NOT NULL,
despatchDate smalldatetime NOT NULL,
returnDate smalldatetime);
```

MODEL b) The ALTER table allows you add extra functionality to a table definition or amend some characteristic of the Table definition (for example change a field width ALTER TABLE X ALTER custname nchar(33))
The SQL for the task of adding referential integrity between Tables **Onloan** and **Copy** is as follows

First add the PRIMARY KEY on Tables Copy and OnLoans (if not done so in MODEL a)

```
ALTER TABLE Copy ADD PRIMARY KEY (CopyID);
ALTER TABLE OnLoan ADD PRIMARY KEY (CopyID,custID,despatchDate);
```

Then add the Foreign key reference

```
ALTER TABLE OnLoan ADD FOREIGN KEY (CopyID) REFERENCES Copy (CopyID);
```

MODEL c)
To list the **dvdName** and **dvdGenre** of DVD's with dvdGenre = Animation that were ranked amongst the top 5 rentals last week and are still ranked in the top 5 for this week you need to write this SQL code :-

```
SELECT dvdName,  dvdGenre
FROM DVD as d, TopTenrentalsThisweek as t10TW,
TopTenrentalsLastweek as t10LW
WHERE d.dvdID = t10TW.dvdID
```

```
AND d.dvdID = t10LW.dvdID
AND d.dvdGenre= 'Animation'
AND t10TW.topTenNo <6
AND t10LW.topTenNo <6
```

Please state the answer thus:-

```
dvdName                                           dvdGenre
------------------------------------------------- ----------
Curse of the Were Rabbit                          Animation
Corpse Bride                                      Animation
```

MODEL d) The Table **TopTenRentalsThisWeek** could be derived from existing data rather than be persisted as a Table. This is how you would achieve this in SQL.

```
select COUNT(o.copyiD) as Count, dvdID
FROM Copy as C, Onloan as O
WHERE c.copyid = o.copyID
AND despatchDate BETWEEN '2006-2-21' AND '2006-2-28'
GROUP BY dvdID
ORDER BY Count DESC

The result set generates the count for ranking and the dvdID.
That is all that is required at this level. The resultset can
be subsequently ordered, but the answer for the data sample
supplied is as follows:

Count       dvdID
----------- -----
5           20467
2           00601
```

Examiner's Comments

MODEL a)
MODELially correct solutions did gain some marks, but it is important NOT to answer MODEL b) in MODEL a). The examiner is looking to see if you can separate the normal CREATE code transaction into 2 independently executable MODELs.
MODEL a and b)
Many candidates didn't check the logic behind their code and in many cases the code would not have worked because they had not established primary keys before enforcing the foreign key constraint! This means either your code in MODEL a) has primary keys applied or you apply ALTER statements as shown in the answer pointer.
MODELs c and d)
These two MODELs were generally answered poorly showing a lack of practice of writing SQL code. It is very difficult to answer SQL coding questions if candidates have not practised very hard writing SQL code that runs on a server database.
One candidate came up with the perfect solution for MODEL d), in fact more that was expected from the model answer. The perfect solution is to completely generate the ranking values for column 1. This is how it is done in Transact-SQL a programmatic version of SQL that uses temporary tables:

```
DECLARE @TableVar TABLE (i int IDENTITY(1,1), c int,dvd nchar(5))

INSERT @TableVar(c,dvd)
select COUNT(o.copyiD) as Count, dvdID
FROM Copy as C, Onloan as O
WHERE c.copyid = o.copyID
AND despatchDate BETWEEN '2006-2-21' AND '2006-2-28'
GROUP BY dvdID
```

```
ORDER BY Count DESC
```

select i,dvd from @TableVar

* please note the following code is not MODEL of the answer pointers, but is provided to allow candidates to try out the SQL code. The code from /* to */ can be copy and pasted into an edit window of a SQL command interpreter such as ISQLW or SQLServer Express.

```
/*
INSERT INTO DVD VALUES('00601','Groundhog Day','Harold','Ramis','Comedy',1993);
INSERT INTO DVD VALUES('10406','MASH','Robert' ,'Altman','Comedy',1969);
INSERT INTO DVD VALUES('20467','Curse of the Were
Rabbit','Nick','Park','Animation',2005);
INSERT INTO DVD VALUES('77890','Crash','Paul','Haggis','Drama',2004);
INSERT INTO DVD VALUES('00056','Boogie Nights','Paul
Thomas','Anderson','Comedy',1997);
INSERT INTO DVD VALUES('34211','Kati Patang','Shakti','Samanta','Drama',1970);
INSERT INTO DVD VALUES('45609','Corpse Bride','Tim' ,'Burton','Animation',2005);
INSERT INTO DVD VALUES('76213','Ladies in Lavender', 'Charles','Dance','Drama',2004);
INSERT INTO DVD VALUES('88609','A Bout De Souffle',  'Jean
Luc','Goddard','Drama',1959);
INSERT INTO DVD VALUES('00003','Jhankaar Beats','Sujoy','Ghosh','Drama',2003);

INSERT INTO Copy VALUES('00001',      '00601', 'On loan');
INSERT INTO Copy VALUES('00002',      '00601', 'Available');
INSERT INTO Copy VALUES('00003',      '00601', 'Available');
INSERT INTO Copy VALUES('00004',      '20467', 'On loan');
INSERT INTO Copy VALUES('00005',      '20467', 'On loan');
INSERT INTO Copy VALUES('00006',      '20467', 'Damaged');
INSERT INTO Copy VALUES('00007',      '34211', 'Available');
INSERT INTO Copy VALUES('00008',      '34211', 'Available');
INSERT INTO Copy VALUES('00009',      '77890', 'Available');

INSERT INTO OnLoan VALUES('00002',    '1234', '2006-02-20',  '2006-02-26');
INSERT INTO OnLoan VALUES('00003',    '1234', '2006-02-20',  '2006-02-26');
INSERT INTO OnLoan VALUES('00001',    '1237', '2006-02-21',  '2006-02-27');
INSERT INTO OnLoan VALUES('00005',    '1235', '2006-02-21',  '2006-02-27');
INSERT INTO OnLoan VALUES('00004',    '1238', '2006-02-24',  '2006-02-28');
INSERT INTO OnLoan VALUES('00006',    '1238', '2006-02-24',  '2006-02-28');
INSERT INTO OnLoan VALUES('00001',    '1236', '2006-02-27',   NULL)  ;
INSERT INTO OnLoan VALUES('00004',    '1236', '2006-02-27',   NULL)  ;
INSERT INTO OnLoan VALUES('00005',    '1237', '2006-02-28',   NULL)  ;

INSERT INTO TopTenRentalsThisWeek VALUES('20467'); INSERT INTO TopTenRentalsThisWeek
VALUES('77890');
INSERT INTO TopTenRentalsThisWeek VALUES('50976'); INSERT INTO TopTenRentalsThisWeek
VALUES('45609');
INSERT INTO TopTenRentalsThisWeek VALUES('63211'); INSERT INTO TopTenRentalsThisWeek
VALUES('88543');
INSERT INTO TopTenRentalsThisWeek VALUES('90256'); INSERT INTO TopTenRentalsThisWeek
VALUES('20567');
INSERT INTO TopTenRentalsThisWeek VALUES('78453'); INSERT INTO TopTenRentalsThisWeek
VALUES('90087');
INSERT INTO TopTenRentalsLastWeek VALUES('20467'); INSERT INTO TopTenRentalsLastWeek
VALUES('90455');
INSERT INTO TopTenRentalsLastWeek VALUES('87654'); INSERT INTO TopTenRentalsLastWeek
VALUES('77890');
INSERT INTO TopTenRentalsLastWeek VALUES('45609'); INSERT INTO TopTenRentalsLastWeek
VALUES('76213');
INSERT INTO TopTenRentalsLastWeek VALUES('45399'); INSERT INTO TopTenRentalsLastWeek
VALUES('50976');
INSERT INTO TopTenRentalsLastWeek VALUES('10043'); INSERT INTO TopTenRentalsLastWeek
VALUES('00003');
*/
```

Question 6

Refer to **Appendix B** at the end of this paper.

a) Produce an Entity Relationship (ER) diagram that models the discourse and supports the requirements given in **Appendix B**. Include the following in your model:

- Entity Types (these are referenced by the bold typed names in **Appendix B**)
- Relationship Types
- Relationship Degrees (1 to Many for example) **(9 marks)**

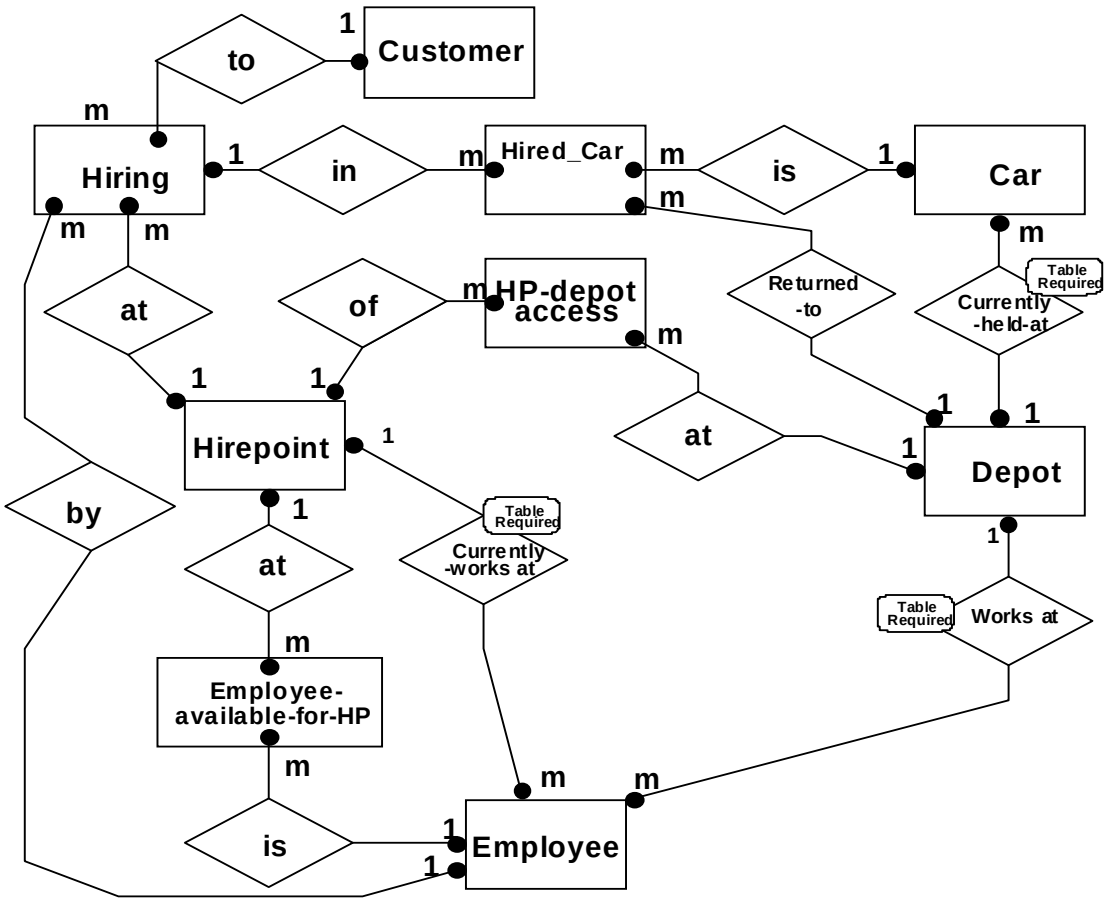
b) Draft out a set of Tables derived from your ER model clearing showing the primary keys, foreign keys and attribute/column names assigned to each table. **(10 marks)**

c) Demonstrate that your Tables can support *AT LEAST THREE* of the requirements specified. **(6 marks)**

You should:
STATE any assumptions made in your modelling and
STATE the diagram notation you have used in part *a)*.

Answer Pointers

a) A generic ER model for the discourse is shown below. There are alternative ways to model the discourse and the examiner considers each model on its merit and consistency with the discourse. Candidates are encouraged to state assumptions so long as they do not contradict the discourse.



Entity Types (these were already identified by the bold typed names in the discourse)
Relationship Types indicated by named diamonds.
Relationship Degrees (1 to m for example). Any many to many relationships have been reduced.

Examiner's Comments on ER Modelling

Generally the standard of data modelling was quite good. However it is difficult to mark some ER models that candidates supply due to inconsistent use of model notations (if a notation is supplied) or no recognised notation supplied at all.

b) It is important to distinguish base tables from the additional tables that could manage the transaction requirements at the end of the discourse. Also note the importance of stating the primary keys, foreign keys and attribute/column names assigned to each table.

Base Tables (neglects hire rate and Relationship Types) could be as follows :-

Car (reg#, car type, colour, HireClass, year, miles_travelled, date _last_service, miles_last_service…..**Object you hire**
Hirepoint (hp#, address, name **… A place where hirings are arranged.**
Depot(depot#, address **… A place where cars are stored / serviced**
Employee (emp#, name, startDate,EndDate**, ….. (Works at Hirepoint or Depot ..)**
Customer (cust#, address **… A person or company who may hire a car**
Hiring (hiring#, date, ….. **A contract to hire a car**

s which
ected to

workplace=ID for Depot or Hirepoint to track history
HPdepotAccess(HP#,Depot#,Location
Hirings(EmployeeID#, sales

MODEL c)
This MODEL of the question from the examiners point of view confirms candidates understanding of how the functional requirements are used in validate an ER model in terms of access paths to the records in mapped tables. The text by ER Howe (Data Analysis for Database Design) shows an example of instance relationships mapped from an ER model. Each instance is a record key value (eg CustomerID=4) and a line connects to a related instance (in another table) for example Carreg# = '123ABC' and then to other entities (eg Depot) consistent with the ER model. The purpose of this diagram is to show the navigation paths through the instances as described above. Successful navigation is an informal way of proving the sufficiency of the ER model.

Examiners Comments on MODELs b) and c)

Generally the marks awarded to both MODELs are dependent on MODEL a) to a large extent. But even if the ER model was insufficient marks were awarded for attempting to use a reasoned mapping approach by considering the degree and MODELicipation constraints. Some candidates clearly used a stated approach but many candidates used their intuition. The only problem with the latter approach was if the mapped tables misrepresented the ER model. It is good practice to state the approach used in the mapping process used from the outsight. MODEL c) also demonstrated a lack of thoroughness when validating an ER modcl to support supplied transactions. It was not stated (like MODEL b) how candidates achieved this validation, rather the examiner expected candidates to use an approach based on their personal/professional experience/practice of validating ER models. In other words candidates knew there was a practical need to do this in their own database designs.