Victoria University of Wellington

DEGREE EXAMINATIONS — 2001 COMP302

MID-YEAR

COMP 302

Database Systems

Time Allowed: 3 Hours

Instructions: Answer all questions.
Make sure your answers are clear and to the point.
Calculators and foreign language dictionaries are allowed.
No reference material is allowed.
There are 180 possible marks on the exam.

CONTENTS:

Question 1. Entity Relationship (ER) Data Model [25 marks]

Question 2. Mapping ER to relational data model [35 marks]

Question 3. SQL and Relational Algebra [30 marks]

Question 4. Functional Dependencies and Normalization [35 marks]

Question 5. Query Optimization [30 marks]

Question 6. Object-Oriented Databases [25 marks]

Question 1. Entity Relationship (ER) Data Model [25 marks]

The basic concepts of a relational database schema are:

- Domain,
- Domain constraints,
- Attribute,
- Relation schema,
- Relation schema constraints (keys, functional dependencies, attribute constraints), and
- Referential integrity constraint.

Draw an ER diagram of the conceptual schema for a database that contains information about the concepts of a relational database schema. The concepts of the relational database schema and their relationships are described in the following manner:

1) Each ***domain*** is identified by a unique *Domain_Name* value, and has additional properties: *Data_Type*, *Max_Length*, *Dom_Default,* *Dom_Value_Range*. These properties constitute the ***domain constraints***.
2) Each ***attribute*** is uniquely identified by a value of *Attr_Name*, and has *Attr_Type* as an additional property. (*Attr_Type* takes values from the set {*stored*, *derived* }).
3) Each ***relation schema*** is identified by a unique *Schema_Name* value, and has the additional properties: *NoOfAttrs*, *NoOfKeys*, *Normal_Form*.
4) A ***key*** is uniquely identified by the name of the relation schema it belongs to, and by the value of its property *Key_No*, i.e. there is an ordinal number associated with each relation schema key. Additionally, every relation schema key has a role, *Key_Role*. (The *Key_Role* property takes values from the set {*primary*, *equivalent* }.)
5) A ***functional dependency*** is uniquely identified by the name of the relation schema it belongs to, and by the value of its property *FD_No*, i.e. there is an ordinal number associated with each relation schema functional dependency. Additionally, every functional dependency has a property *NoOfLHS_Attr*. (It is supposed that functional dependencies have only one attribute on their right hand side.)
6) Every attribute is associated with exactly one domain.
7) One domain can be associated with more than one attribute, but there may be some domains that are not associated with any attribute.
8) Every relation schema is associated with at least one attribute.

9) One attribute can be associated with more than one relation schema, but there may be some attributes that are not associated with any relation schema.

10) The association between the attributes and a relation schema carries the information about relation schema attribute constraints, and has the following properties: *RSA_Null*, *RSA_Default*, *RSA_Value_Range*.

11) Every relation schema key is associated with at least one attribute.

12) One attribute can be associated with more than one relation schema key, but there may be some attributes that are not associated with any relation schema key.

13) A functional dependency has two associations with attributes. One for its left hand side, and the other for its right hand side.

14) The left hand side of a functional dependency contains at least one attribute, whereas its right hand side contains exactly one attribute.

15) An attribute can be associated with more than one functional dependency in the role of the left or right hand side attribute, but there can be some attributes that are not involved in any functional dependency.

16) A ***referential integrity*** constraint is an association between two (not necessarily distinct) relation schemas. In this association, one relation schema takes the role of ***referencing***, and the other takes the role of ***referenced*** relation schema.

17) One relation schema can reference more than one relation schema, but there may be some relation schemas that do not reference any relation schema.

18) One relation schema can be referenced by more than one relation schema, but there may be some relation schemas that are not referenced by any relation schema.

Question 2. Mapping ER to relational data model [35 marks]

Consider the ER that describes a part of a *BANK* database. The diagram is shown in the Figure 2.1 on the next page. Note that the entity type attributes are given separately, in the form

ENTITY_TYPE_NAME (*Atribute*$_1$,…, *Attribute*$_n$),

and that entity type keys are underlined, whereas weak entity type partial keys are dashed underlined.

Each bank can have multiple branches, and each branch can have multiple accounts and loans. The accounts and loans belong to customers.

a) Map conceptual schema *BANK* to a set of relation schemas. (You will be asked to define referential integrity constraints of the relational database schema *BANK* in question 2.(e).) [15 marks]

b) Check whether the set of relation schemas you produced in question 2.(a) is in BCNF (Boyce_Codd normal form) with the regard to the set of functional dependencies implied by the relation schema keys. If not, transform it into BCNF. [1 mark]

c) Check whether the set of relation schemas you produced in the question 2.(b), represents a lossless join decomposition. If not, transform it into a lossless join decomposition relation schema set, that will remain in BCNF. [5 marks]

d) For each relation schema you produced in question 2.(c), and for every foreign key it contains, designate the value of the constraint

Null (*relation_schema_name*, *foreign_key*).

Let the constraint value be *N* if the foreign key is not allowed to contain null value, and *Y* otherwise. [7 marks]

e) Write the referential integrity constraints for the set of relation schemas you produced in the question 2.(c). [7 marks]

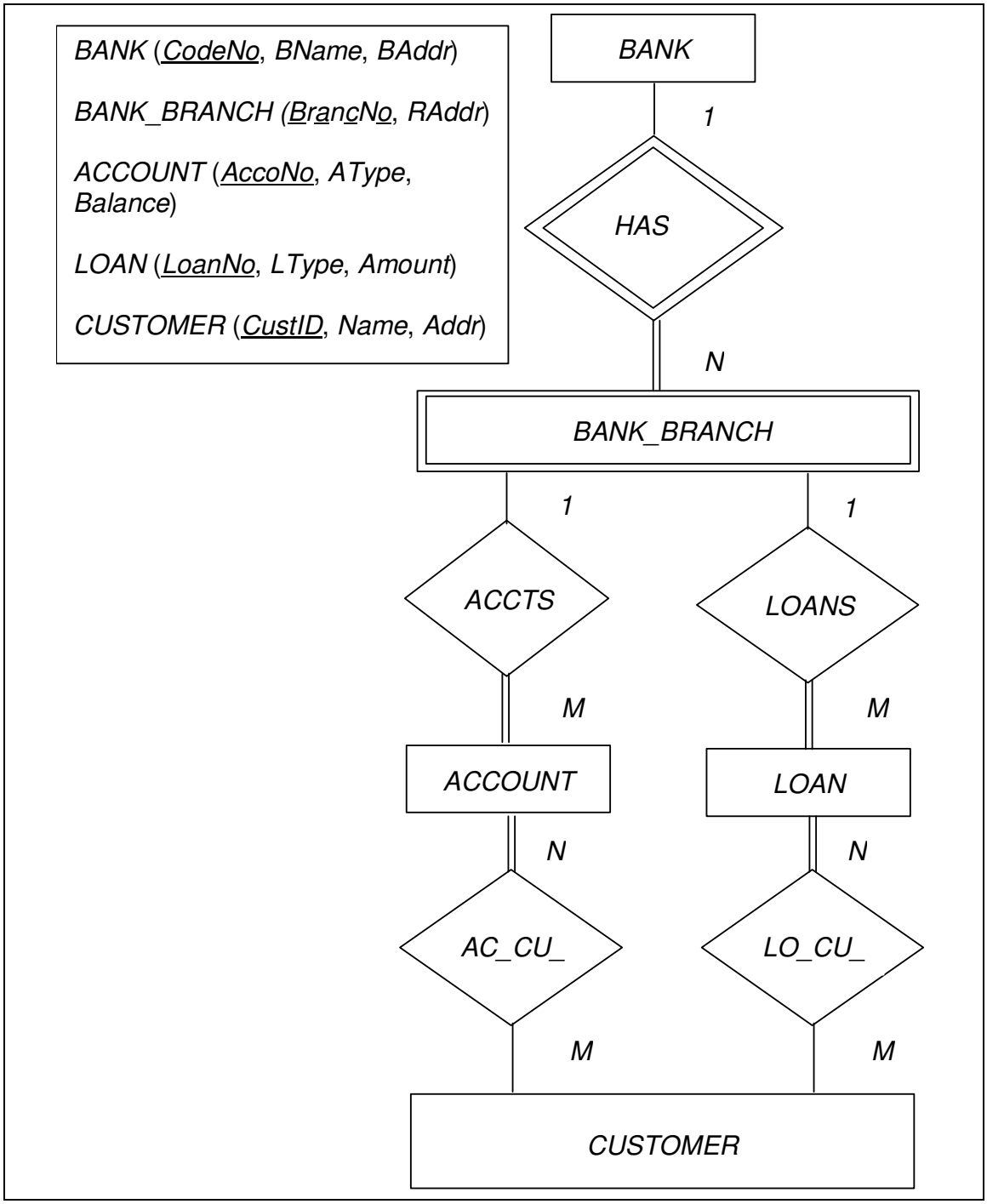
BANK (CodeNo, BName, BAddr)
BANK_BRANCH (BrancNo, RAddr)
ACCOUNT (AccoNo, AType, Balance)
LOAN (LoanNo, LType, Amount)
CUSTOMER (CustID, Name, Addr)
BANK
1
HAS
N
BANK_BRANCH
1
1
ACCTS
LOANS
M
M
ACCOUNT
LOAN
N
N
AC_CU_
LO_CU_
M
M
CUSTOMER

Figure 2.1

Question 3. SQL and Relational Algebra [30 marks]

Consider the following set of relation schemas for a database named *AUTO_SALES* that keeps track of car sales in a car dealership. The relation schema *Options* refers to some optional equipment installed on a car.

Car({*Serial_No*, *Model*, *Manufacturer*, *Price*}, {*Serial_No*, *Model*+ *Manufacturer*}),

Options({*Serial_No*, *Option_No*, *Option_Price*}, {*Serial_No*+ *Option_No*}),

Sales({*Serial_No*, *SalesPersonID*, *Sale_Price*}, {*Serial_No*+ *SalesPersonID*}),

SalesPerson({*SalesPersonID*, *Name*, *Phone*}, {*SalesPersonID*}),

a) Write the following SQL queries on the *AUTO_SALES* database:

I. Retrieve the serial number, manufacturer, and sale-price for all the cars sold by sales person *James*, whose sales person id is *007*. [5 marks]

II. Retrieve the serial number and the model of all the cars that have no options. [5 marks]

III. Retrieve the model, the manufacturer, and the total number sold for the cars that have been sold in the greatest number. [10 marks]

b) Write the following relational algebra queries on the *AUTO_SALES* database:

I. Retrieve the serial number, manufacturer, and sale-price for all the cars sold by sales person *James*, whose sales person id is *007*. [5 marks]

II. Retrieve the serial number and the model of all the cars that have no options. [5 marks]

Question 4. Functional Dependencies and Normalization [35 marks]

a) Express the following real world constraints using functional dependencies: [7 marks]

I. Every office (*Office*) has its own extension phone number (*ExPhNo*).

II. Every extension phone number (*ExPhNo*) belongs to at most one office (*Office*).

III. Every student (represented by *StudentID*) can get at most one grade (*Grade*) for a paper (represented by *PaperID*) in the same term (*Term*) of a year (*Year*).

IV. Two different book titles (*Title*) can't have the same *ISBN* number (*ISBN*).

b) Consider the universal relation schema, defined by the following attribute and functional dependency sets: [28 marks]

$$U = \{A, B, C, D, E, F, G, H, I, J\},$$

$$F = \{BC \rightarrow A,\ AD \rightarrow E,\ E \rightarrow F,\ C \rightarrow B,\ BC \rightarrow D,\ BC \rightarrow F,\ FG \rightarrow H,\ FG \rightarrow I,\ E \rightarrow G,\ I \rightarrow G\}.$$

I. Normalize the relation schema (U, F) by the applying synthesis algorithm.

II. Check whether the relation schema set, you obtained in question 4.b(I) is attribute and dependency preserving.

III. What normal form the relation schema set you obtained in question 4.b(I) is in? Explain your answer.

Question 5. Query Optimization [30 marks]

Consider the description of a part of a database schema, given in the Table 5.1, and the corresponding parameters of the physical database structure, given in the Table 5.2. In the Table 5.2, *L* is the tuple length (size), *f* is the blocking factor, *B* is the block size, and *x* is either the primary, or secondary key index height. The values of all the Table 5.2 parameters pertain to the base relations.

Attribute	Data Type	Max. Length	Null	Default
Relation name: *Customer*, Primary Key: *CustomerID*				
CustomerID	Integer	4	N	
Lname	Char	15	N	
Fname	Char	15	Y	
City	Char	15	N	
Relation name: *Cust_Order*, Primary Key: *OrderID*				
OrderID	Integer	4	N	
OrderDate	Date	3	Y	
CustomerID	Integer	4	N	

Table 5.1.

Relation schema	*L*	*f*	*B*	*r*	*x* (primary)	*x* (secondary)
Customer	*49*	*10*	*490*	*1300*	*2*	*1*
Cust_Order	*11*	*45*	*495*	*26000*	*3*	*2*

Table 5.2.

Suppose:

- Customers are from 13 different cities, and there is an approximately equal number of customers from every city.
- Every customer has made approximately the same number of orders.
- There is a buffer pool of 2500 bytes provided for query processing in the main memory.
- The intermediate results of query evaluation are materialized,
- The final result of the query is materialized, and
- The size of each intermediate result block should not exceed 500 bytes.

a) Draw the heuristic optimization tree that corresponds to the SQL query:

```
SELECT co.OrderID, c.CustomerID, c.FName
FROM Cust_Order co, Customer c
WHERE co.CustomerID = c.CustomerID
ORDER BY CustomerID, OrderID;
```

[5 marks]

b) Suppose there are no indexes on relations *Customer*, and *Cust_Order*. State any assumptions you have made, and calculate the cost of executing the query:

```
SELECT co.OrderID, c.CustomerID, c.FName
FROM Cust_Order co, Customer c
WHERE co.CustomerID = c.CustomerID
ORDER BY CustomerID, OrderID;
```

[10 marks]

c) Suppose:
- there are primary key indexes on relations *Customer* and *Cust_Order*,
- there is a secondary index on *City* in the relation *Customer*, and
- there is a secondary index on *CustomerID* in the relation *Cust_Order*.

State any assumptions you have made, and calculate the cost of executing the query:

```
SELECT co.OrderID, c.CustomerID, c.FName
FROM Cust_Order co, Customer c
WHERE co.CustomerID = c.CustomerID AND c.City = “Wellington”
ORDER BY CustomerID, OrderID;
```

[15 marks]

Note:

There are some of the formulae you may need when computing the estimated query cost given in the attachment to this exam document.

Question 6. Object-Oriented Databases [25 marks]

a) Use Objectivity DDL to define a small object-oriented database that contains data about: employees, projects, and their associations. [5 marks]

- Every employee has a unique name, a position, and an association with the projects he/she works on.
- Every project has a unique project name, a budget, and an association with the employees that work on the project.
- One employee can work on more than one project, and more than one employee can work on the same project.

b) Suppose there are two employee and two project objects created. These are:

- (*James*, *DBA*), (*Susan*, *Programmer*), and
- (*UniversityDatabase*, *100000*), (*ElectronicSubmission*, *25000*).

Use objects of Objectivity special classes to write a C++ program, that will associate both of the two created employee objects with both of two created project objects. [10 marks]

c) After associating the given objects, destroy the association between the *James* object and the *UniversityDatabase* object. [10 marks]

Note:

You are not expected to:

- Initialize the database system, or
- Open Objectivity persistent objects.

Minor errors in syntax will not be penalized.

****

Attachment to COMP 302 Degree Examination – 2001

FORMULAE FOR COMPUTING QUERY COST ESTIMATE

Blocking factor: $f = \lfloor B / L \rfloor$

Number of blocks: $b = \lceil r / f \rceil$

Selection cardinality of the attribute A : $s(A) = r / d(A)$

C (project) = $b_1 + b_2$

C (project_sort) = $b_1 + b_2(3 + 2\lceil log_2 b_2 \rceil)$

C (select_linear) = $b_1 + \lceil s / f \rceil)$

C (select_sec_key) = $x + s + \lceil s / f \rceil)$

C (nested_join) = $b_1 + b_2 \lceil b_1 / (n - 2) \rceil + \lceil js * r_1 * r_2 / f \rceil)$

C (single_join) = $b_1 + r_1 * f(index_2) + \lceil js * r_1 * r_2 / f \rceil)$

C (sort_join) = $b_1(3 + 2\lceil log_2 b_1 \rceil) + b_2(3 + 2\lceil log_2 b_2 \rceil) + \lceil js * r_1 * r_2 / f \rceil)$

C (sort) = $2b(1 + \lceil log_2 b \rceil)$

f (*index*):

primary index: $f(index) = x + 1$

secondary index: $f(index)$: $x + s$