Name: ____________________

Department of Computer Science

Computer Science 3319a
Second Test, Nov. 19, 2009
Time allowed: 50 minutes

Prof. S. Osborn

No aids (including electronic aids)

Answer all questions on the exam paper.

This exam consists of 10 pages; the last page can be used for rough work but must be turned in.

Name: ____________________

Student Number ____________________

Question	Maximum	Your Mark
1	8	
2	15	
3	5	
4	6	
5	18	
6	10	
Total	62	

1. Consider the following relations with data:

R	A	B	C	D	E
	a	z	1	m	1
	b	x	1	m	6
	a	y	2	p	5
	b	z	2	p	6
	a	y	3	n	3

S	E	G	H
	6	1	a
	5	3	b
	2	5	c

T	H	I	J
	a	x	2
	a	z	2
	d	w	4

(a) (4 marks) Show the relation that results from execution of the following domain calculus query:

{by | (∃ a) (∃ c) (∃ d) (∃ e) (R(abcde) and (∃ l) (∃ m) (∃ n)(S(lmn) and (∃ x) (∃ z) (T(xyz) and e = l and n = x and z = 2))) }

	B	I
	x	x
	x	z
	z	x
	z	z

2 marks for having 2 columns with right headings, 2 marks for data

(b) (4 marks) Show the relation that results from this tuple relational calculus query:

{ r.B, s.G, | R(r) and S(s) and (∃ t) (T(t) and r.E = s.E and s.H = t.H and (t.J = 4 or r.D = ”p”)) }

	B	G
	z	1

2 marks for having 2 columns with right headings, 2 marks for data

2. Consider the following relations for a relational database:

   Customer(Name, Address, Age, Occupation)

   Item(ManuName, ItemName, Price)

   Likes(Name, ManuName, ItemName, Rank)

   (a) (5 marks) Give a **Tuple Relational Calculus** query to find, for items made by Sony, the names of all customers who like something whose price is greater than 1000. Put the customer name and the item name in the answer.

   { m.Name, m.ItemName | (∃ i)(Likes(m) and Item(i) and
   m.ManuName = "Sony" and i.Price > 1000 and
   i.ManuName = m.ManuName and i.ItemName = m.ItemName) }

   1 mark for correct answer attributes, 1 for ∃, 1 for relation predicates, 1 for join predicates and 1 for query predicates.

   (b) (5 marks) Give a **Domain Relational Calculus** query to find, for customers whose age is greater than 18, the items they like and have given a rank of 1. Put the customer name and the item name in the answer.

   { a, n | (∃ b)(∃ c)(∃ d)(∃ o)(∃ p)(∃ q)
   (Customer(abcd) and Likes(nopq)
   and c > 18 and q = 1
   and a = n) }

   1 mark for correct answer attributes, 1 for ∃ of all the domain variables, 1 for relation predicates, 1 for join predicate(s) and 1 for query predicates.

Here are the relations again.
Customer(Name, Address, Age, Occupation)
Item(ManuName, ItemName, Price)
Likes(Name, ManuName, ItemName, Rank)

(c) (5 marks) Give a **Tuple Relational Calculus** query to find all customer names such that the customer likes **ALL** the items liked by "George".

GeorgeLikes = { m.ManuName, m.ItemName | Likes(m) and m.Name = "George" }
Answ = { c.Name | Customer(c) and (∀ g)(∃ m)(GeorgeLikes(g) and Likes(m) and
g.ManuName = m.ManuName and g.ItemName = m.ItemName and c.Name = m.Name)}
1 mark for customer names in the answer, 2 for getting ∀ to work, 2 for connecting that to the answer.

3. (5 marks) Consider the relations again from the previous question:

Customer(<u>Name</u>, Address, Age, Occupation)

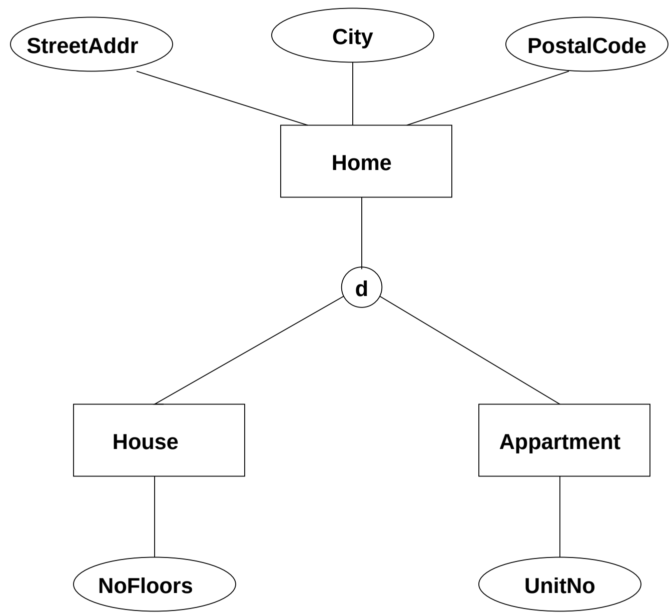
Item(<u>ManuName, ItemName</u>, Price)

Likes(<u>Name, ManuName, ItemName</u>, Rank)

Which of the following functional dependencies are reasonable FDs? Just write **yes** or **no** beside the potential FD.

(a) Name → Age – yes
(b) ItemName → Price – no
(c) ItemName → Rank – no
(d) ManuName, ItemName → Rank – no
(e) Occupation → Name – no

4. (6 marks) Consider the following EER diagram:



(a) (2 marks) What does the "d" in the small circle above mean?

disjoint subclasses

(b) (4 marks) Give relation scheme(s) for this diagram. Explain **briefly** why you chose to do it the way you did.

House(StreetAddr, City, PCode, NoFloors)

Apartment(StreetAddr, City, PCode, UnitNo)

is one solution. Others were accepted.

2 marks for getting all attributes somewhere, 1 for having one of the 4 possible solutions, 1 for why.

5. Consider the following function dependencies for a relation R(A, B, C, D, E):

   AC → DE

   DE → C

   A → B

   (a) (3 marks) Is DE a candidate key for this relation with these functional dependencies? Justify your answer.

   $(DE)^+$ = DEC. Therefore it is not a key.

   (b) (4 marks) Give a derivation of ADE → ABCDE from the given functional dependencies above. Show how you got your answer using an algorithm or the inference rules for FDs.

   ADE → ADE by reflexivity

   use DE → C in the algorithm, because DE is included in ADE, can conclude that ADE → ACDE

   use A → B, gives ADE → ABCDE

   An alternative derivation using the inference rules directly:

   1. ADE → ADE by reflexivity
   2. ADE → DE by reflixivity, and DE → C is given, therefore ADE → C by transiticity
   3. therefore ADE → ACDE by union of 1 and 2
   4. ADE → A by reflixivity and A → B is given, therefore ADE → B by transitivity
   5. therefore ADE → ABCDE by union of 3 and 4.

   (c) (3 marks) Can any attributes be removed from ADE and still have it be a key? Justify your answer.

   no. $(AD)^+$ = ABD

   $(AE)^+$ = ABE

   $(DE)^+$ = CDE

   none of these = ABCDE.

Here are the FDs again: AC → DE   DE → C   A → B

(d) (2 marks) Is R(A, B, C, D, E) with the above functional dependencies in second normal for? If not, explain why.

no, because ADE is a key, and A → B is a non-full dependency

(e) (2 marks) Is R(A, B, C, D, E) with the above functional dependencies in third normal form? If not, explain why.

no, becuause it is not in 2NF.
or no, because there is a transitive dependency ADE → A → B.

(f) (2 marks) Is R1(A, C, D, E), R2(A, B) a lossless join decomposition? Justify your answer.

yes, because the intersection of the two relations is A, which is a key for R2.

(g) (2 marks) Is R1(A, C, D, E), R2(A, B) a dependency preserving decomposition? Justify your answer.

yes, because A → B is contained in R2, and the other FDs are contained in R1, so there is a place to verify all the FDs in this set of tables.

6. (10 marks) This deals with the example from a previous midterm: N stands for Driver Name, D for Driver's Licence Number, A for Address, P for Phone, C for City, M for manufacturer, T for car type, L for Car Licence Plate Number, and Q for Date Acquired. We have the following functional dependencies:

   $D \rightarrow N$

   $D \rightarrow A$

   $D \rightarrow P$

   $P \rightarrow C$

   $ND \rightarrow C$

   $LT \rightarrow M$

   $LD \rightarrow Q$

   $LDN \rightarrow Q$

   Find a minimal covering of these functional dependencies. Then use it to synthesize a set of dependency preserving, 3NF relations with a lossless join. I.e. use the modified algorithm 11.2 from the notes.

   $D^+ = DNAPC$

   $P^+ = PC$

   $(ND)^+ = DNACP$

   $(LT)^+ = LTM$

   $(LD)^+ = LDNAPCQ$

   $(LDN)^+ = LDNAPCQ$

   Minimal covering gets rid of $ND \rightarrow C$, and $LDN \rightarrow Q$

   Output is:

   R1($\underline{D}$, N, A, P)

   R2($\underline{P}$, C)

   R3($\underline{L, T}$ M)

   R4($\underline{L, D}$, Q)

   add R5($\underline{L, D. T}$) to give the lossless join.