EECS-3421m: Midterm Test

Electrical Engineering & Computer Science

Lassonde School of Engineering

York University

Family Name: ______________________________

Given Name: ______________________________

Student#: ______________________________

EECS Account: ______________________________

Instructor: Parke Godfrey
Exam Duration: 75 minutes
Term: Winter 2018
Instructions

- **rules**
  - The test is closed-note, closed-book. Use of a calculator is permitted.
- **answers**
  - Should you feel a question needs an assumption to be able to answer it, write the assumptions you need along with your answer.
  - If you need more room to write an answer, indicate where you are continuing the answer.
  - For multiple choice questions, choose *one* best answer for each of the following. There is no negative penalty for a wrong answer.
  - For E/R diagrams, keep the elements as *simple* as possible; e.g., an attribute is simpler than a relationship which is simpler than an entity.
- **notation**
  - For schema, the underlined attributes indicate a table's primary key (and are not nullable). Attributes with an "*" are not nullable. Foreign keys are indicated by FK.
  - Assume *set* semantics for relational-algebra expressions.
- **points**
  - Each question is marked by the number of points it is worth.
  - There are five major parts worth 10 points each, for 50 points in total.

Marking Box	
1.	/10
2.	/10
3.	/10
4.	/10
5.	/10
Total	/50

1. [10pt] **Relational Schema.** *A get-rich scheme.* [Exercise]

(a) [4pt] York University has struck it rich with endowments! As part of the windfall, a *student* enroled in a *class may* have a *tutor* assigned to him or her for help with that class. Dr. Dogfury diagrammed this in E/R as in Figure 1.

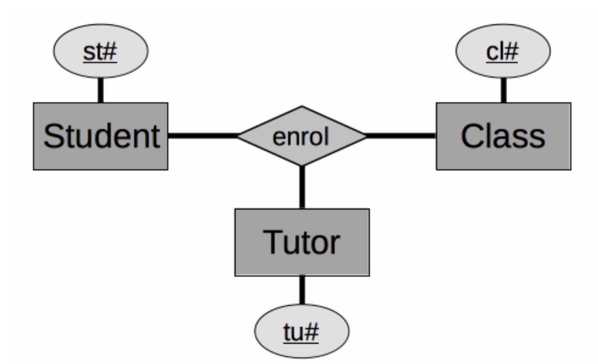
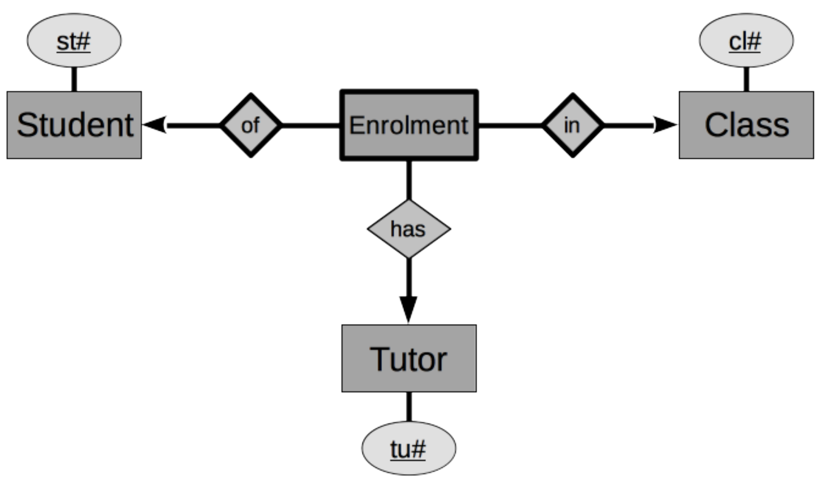


Figure 1: Dr. Dogfury's E/R for *enrol* with *Tutor*.

You say, however, that is wrong, because *not every* student in a class has to have a tutor assigned. (It should be instead *zero* or *one*.)
Draw a correct E/R schema for this.



+2 reasonable solution
+1 elements correct (e.g., Enrolment as a connecting weak entity set)
+1 everything correct; to Tutor is 0/1
2 if original but with 0/1 arrow;
1 1 if with exactly one arrow.

For Questions 1b & 1c, use the shorthand notation as in Figure 5 on page 13.

(b) [3pt] Translate Dr. Dogfury's E/R schema in Figure 1 from Question 1a into a relational schema.

Student(*st#*)
Class(*cl#*)
Tutor(*tul#*)

Enrol(*st#, cl#, tu#*)
FK (st#) refs **Student**
FK (cl#) refs **Class**
FK (tu#) refs **Tutor**

+1 the four tables
+1 "primary" keys are right
+1 the three FKs of Enrol

(c) [3pt] Translate the E/R schema in Figure 2 into a relational schema.

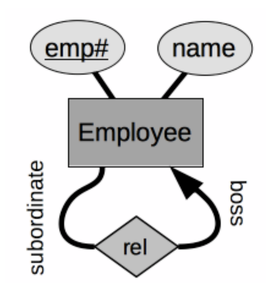


Figure 2: Employee entity set with boss relationship set.

Employee(*emp#, name, boss*)
FK (boss) refs **Employee** *(emp#)*

+1 has right attributes, including boss
+2 FK is cast correctly; boss *is nullable*
in two table solution, FKs right, etc.

2. [10pt] **Entity Relationship.** *You haven't met my relations!* [ANALYSIS]

For this Question, consider the Philatelist Schema given in Figure 5 on page 13 (detachable).

The following is just further explanation about the schema. A *house* is an auction house. The stamps in the database are rare and valuable, so they are tracked per individual stamp. The auction houses own stamps to auction. The price with a stamp is the base price for its auction. Listing provides the "book" price for stamps of a given series in a given condition. And an assessment done by an assessor of a stamp gives an opinion about it (a recommended price and description).

(a) [3pt] Does the house where the assessor of a stamp works and the house that owns the stamp have to be the same, according to the schema's logic?

Why or why not?

> *No. Stamp is not weak on House (via owns). So Assessment does not restrict the value of the House's title of the Assessor's workplace (part of Assessor's key) since the stamp's owning House title is not seen.*

(b) [2pt] In the present schema, we can only store a single assessment by given assessor for a given stamp.

Provide a simple change that you could make to the *relational schema* that would change this so that we could store more than one assessment by given assessor for a given stamp.

(Do *not* provide a modified E/R diagram! Provide a modified *relational schema*. Again, use the shorthand notation as in Figure 5 on page 13.)

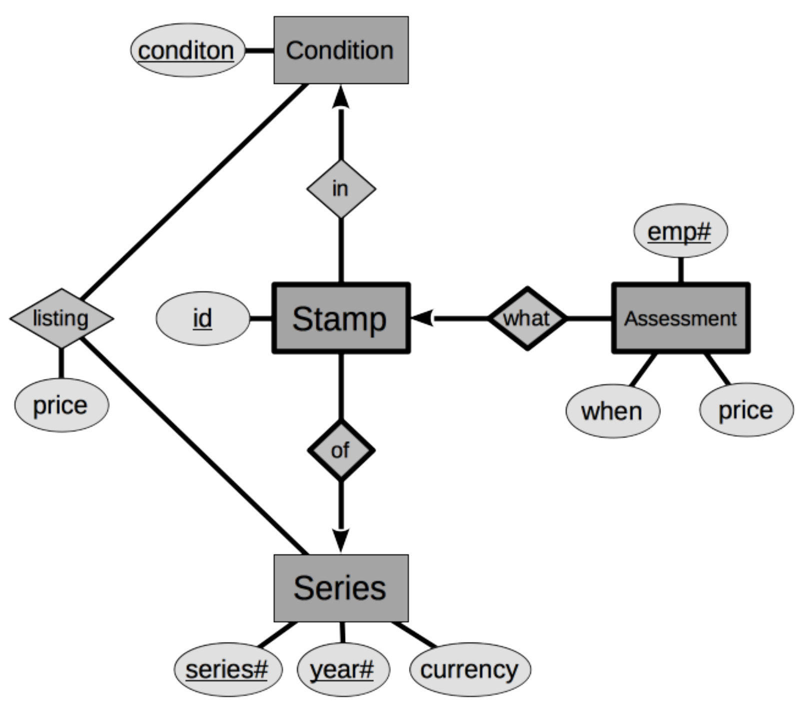
> *We can "weaken" the key of Assessment to provide this, by adding an additional attribute to the key. Adding* when *to the key is a good candidate.*

(c) [5pt] Reverse engineer the relational schema in Figure 3 into an E/R diagram that captures the logic of the schema.

Series(series#, year, currency)
Condition(condition)
Stamp(id, series#, year, condition∗, price)
FK (series, year) refs **Series** — *of*
FK (condition) refs **Condition** — *in*
Listing(series#, year, condition, price)
FK (series, year) refs **Series** — *for*
FK (condition) refs **Condition** — *at*
Assessment(emp#, id, series#, year, when, price)
FK (id, series, year) refs **Stamp** — *what*

Figure 3: Simplified Philatelist Schema.

For this Question, *ignore* the full schema—and any extra complications therein—as given in Figure 5. For notation for hand-drawing an E/R, use the *E/R diagram hand-drawing guide* given in Figure 6 on page 14. Keep the elements as *simple* as possible; e.g., an attribute is simpler than a relationship which is simpler than an entity.



+2 cover reasonably all the elements
+2 things are weak that should be; keys are right
+1 listing is a rel-ship

3. [10pt] **General.** *Much choice!* [Multiple Choice]

(a) [1pt] *Data independence* is that

A. all information in the database is to be represented in one and only one way, namely by values in column positions within rows of tables.
B. all views that are theoretically updatable must be updatable by the system.
$\boxed{\textbf{C.}}$ changes that are made to the physical storage representations or access methods must not require changes be made to application programs.
D. changes that are made to tables that do not modify any of the data already stored in the tables must not require changes be made to application programs.
E. data in different tables must not be related.

(b) [1pt] In E/R, a *one-one* relationship set

A. is not allowed.
B. is an E/R construct that cannot be expressed in a relational schema.
C. can always be replaced in a logically equivalent way by two *one-many* relationship sets.
D. may only (recursively) relate the same entity set to itself.
$\boxed{\textbf{E.}}$ is rare, but is sometimes logically needed for the domain being modelled.

(c) [1pt] In E/R, a *connecting* weak entity set

A. is an E/R construct that cannot be expressed in a relational schema.
B. does not take its keys from other entity sets as does a "regular" weak entity set.
C. is weak only on one other entity set.
D. is logically equivalent to a *sub-entity set* (via an *isa* hierarchy).
$\boxed{\textbf{E.}}$ is logically equivalent to a *relationship set*.

(d) [1pt] Consider a relation **R** with five attributes: A, B, C, D, and E. Attribute C never appears on the right-hand side of any non-trivial functional dependency applying to **R**. How many different possibilities are there for what a candidate key for **R** can be?

A. 1
B. 4
C. 5
$\boxed{\textbf{D.}}$ 15
E. 31

(e) [1pt] Why are the normal forms useful?

$\boxed{\textbf{A.}}$ By having a relational schema in a given normal form, it guarantees that certain types of data anomalies cannot occur.
B. They help us find anomalies in the data.
C. They are just a tool for checking whether our relational design makes sense or not.
D. If the schema is in BCNF, we are guaranteed that queries will execute faster than if it were not in BCNF.
E. They are useless, but earn database consultants lots of money. (Don't tell anyone!)

(f) [1pt] The "BC" in BCNF stands for

A. *backward compatible*
B. *better consistency*
C. Backus Church
$\boxed{\textbf{D.}}$ Boyce Codd
E. Burns Crosby

(g) [1pt] With respect to a set of (prescribed) functional dependencies $\mathcal{F}$ over the set of attributes $\mathcal{A}$,

A. there must exist a BCNF schema that is *dependency preserving*, but it can be computationally *intractable* to find.
$\boxed{\textbf{B.}}$ there must exist a 3NF schema that is *dependency preserving*, but not necessarily a BCNF schema that is *dependency preserving*.
C. it is not guaranteed that there exists a 3NF schema that is *dependency preserving*.
D. it is not guaranteed that there exists a 2NF schema that is *dependency preserving*.
E. all schema are *dependency preserving*, by definition.

(h) [1pt] Consider a prescribed functional dependency $\mathcal{X} \mapsto \mathsf{A}$ with respect to relation $\mathbf{R}$. We know that there is an attribute $\mathsf{B} \in \mathcal{X}$ such that B is *not* prime.

A. $\mathbf{R}$ is *not* in 2NF.
$\boxed{\textbf{B.}}$ $\mathbf{R}$ is in 2NF, but might not be in 3NF.
C. $\mathbf{R}$ is in 3NF, but might not be in BCNF.
D. $\mathbf{R}$ is in 2NF and in BCNF, but is *not* in 3NF.
E. *None of the above is necessarily true in all cases.*

(i) [1pt] Consider the relations $\mathbf{R}(\underline{\mathsf{A}}, \mathsf{B}*)$ and $\mathbf{S}(\mathsf{A}*, \underline{\mathsf{B}})$, where $\mathbf{R}$ has a foreign key referencing $\mathbf{S}$ via B, and $\mathbf{S}$ has a foreign key referencing $\mathbf{R}$ via A.

Which of the following is guaranteed to produce fewer than, or at most the same, number of tuples as any of the others?

$\boxed{\textbf{A.}}$ $\mathbf{R} \bowtie \mathbf{S}$
B. $\mathbf{R} \cup \mathbf{S}$
C. $\mathbf{R} \bowtie \pi_{\mathsf{B}}(\mathbf{S})$
D. $\pi_{\mathsf{A}}(\mathbf{R}) \bowtie \mathbf{S}$
E. *There is not enough information to answer this.*

(j) [1pt] Consider the relations $\mathbf{R}(\underline{\mathsf{A}}, \underline{\mathsf{B}})$, $\mathbf{S}(\underline{\mathsf{B}}, \underline{\mathsf{C}})$, and $\mathbf{T}(\underline{\mathsf{C}}, \underline{\mathsf{A}})$.

One of these is not like the others. That is, one can evaluate differently than the other four. Which one?

A. $\pi_{\mathsf{A,B}}((\mathbf{R} \bowtie \mathbf{S}) \bowtie \mathbf{T})$
B. $\pi_{\mathsf{A,B}}(\mathbf{R} \bowtie (\mathbf{T} \bowtie \mathbf{S}))$
C. $\mathbf{R} \bowtie \pi_{\mathsf{A,B}}(\mathbf{S} \bowtie \mathbf{T})$
D. $\pi_{\mathsf{A,B}}(\mathbf{R} \bowtie \mathbf{S}) \bowtie \pi_{\mathsf{A,B}}(\mathbf{S} \bowtie \mathbf{T})$
$\boxed{\textbf{E.}}$ $\pi_{\mathsf{A,B}}(\mathbf{R} \bowtie \mathbf{T}) \bowtie \pi_{\mathsf{A,B}}(\mathbf{R} \bowtie \mathbf{S})$

---

4. [10pt] **Design Theory.** *Which way to the quay?* [Analysis]

(a) [7pt] Consider the relation **T** with attributes A, B, C, and D, and with the following functional dependencies (FDs):

$$\mathsf{A} \mapsto \mathsf{B} \qquad \mathsf{D} \mapsto \mathsf{A}$$
$$\mathsf{BC} \mapsto \mathsf{D}$$

i. [2pt] What are the keys of **T**? AC, BC, & CD

Consider normal forms 2NF < 3NF < BCNF in that precedence order. For Questions 4(a)ii to 4(a)iv, state the lowest normal form— e.g., 2NF is lower than 3NF— that the FD violates, or say *none* if it violates none.

ii. [1pt] $\mathsf{A} \mapsto \mathsf{B}$ BCNF

iii. [1pt] $\mathsf{D} \mapsto \mathsf{A}$ BCNF

iv. [1pt] $\mathsf{BC} \mapsto \mathsf{D}$ none

v. [2pt] Construct a dependency preserving, lossless-join BCNF decomposition of **T**.

$\underline{BC}D$, $A\underline{D}$, and $\underline{A}B$.
*We can obtain $\underline{BC}D$ and $A\underline{D}$ by decomposing **T** losslessly with $D \mapsto A$ which violates BCNF. Nothing violates BCNF for these. But we no longer cover $A \mapsto B$. Thus, we can* add *$\underline{A}B$ as a table which will cover this, and which is in BCNF in itself.*

(b) [3pt] Give an example of a *deletion anomaly* that can occur because of a *transitive dependency.*

Consider relation $\mathbf{R}$ *with* $E \mapsto NJW$ *and transitive dependency* $J \mapsto W$. *Say we had values* $J =$ 'boss' *and* $W = 60$ *appear in only* one *tuple in* $\mathbf{R}$, *say, for* $E =$ 'Anne'. *If the tuple for* $E =$ 'Anne' *is deleted, we have lost that* $J =$ 'boss' $\mapsto W = 60$.

Extra Space

5. [10pt] **Relational Algebra.** *They said I'd never use algebra!* [EXERCISE]

(a) [3pt] Consider the Colours schema in Figure 4 on page 13 (as used in class). Write a *relational-algebra* expression to show products by prod# and pname that are available in the colour *pink*.

$$\pi_{\mathit{prod\#,pname}}(\textbf{Product} \bowtie \sigma_{\mathit{colour='pink'}}(\textbf{Avail\_Colours}))$$

+1 the '⋈'
+1 the 'σ'
+1 the 'π'

(b) [2pt] Consider the Philatelist schema in Figure 5 on page 13. Write a *relational-algebra* expression that accomplishes the following. List each auction house's country as *site* with the series' country as *origin* such that the house owns stamps in *mint* condition from that series.

$$\pi_{\mathit{site,origin}}(\pi_{\mathit{title,country \rightarrow site}}(\textbf{House}) \bowtie \pi_{\mathit{title,country \rightarrow origin}}(\sigma_{\mathit{condition='mint'}}(\textbf{Stamp})))$$

+1 proper '⋈' and sources
+1 renaming of attr's and 'σ'

A	B
1	2
2	3
3	4

R

B	C
1	2
2	3
3	4

S

C	A
1	2
2	3
3	4

T

Consider the three tables **R**, **S**, & **T** above for Questions 5c, 5d, & 5e.

(c) [2pt] Show the result of **R** $\bowtie$ **S**.

A	*B*	*C*
1	*2*	*3*
2	*3*	*4*

2 all correct
1 missing tuple or extra tuple
0 otherwise

(d) [2pt] Show the result of **R** $\bowtie$ **T**.

A	*B*	*C*
2	*3*	*1*
3	*4*	*2*

2 all correct
1 missing tuple or extra tuple
0 otherwise

(e) [2pt] Show the result of **R** $\bowtie$ (**S** $\bowtie$ **T**).

A	*B*	*C*

+1 empty tuple set
+1 correct schema of table returned
(important it was clear a table is returned, just that it has no tuples)

Yes, this question adds up to eleven points! I cannot add. So scored out of eleven.

Extra Space

Relax. Turn in your test. Return to the wild!

Reference *(Detach this page for convenience, if you want.)*

Schema for the Colours Database.

Customer	
cust#	PK
cname	
fav_colour	
phone#	

Item	
item#	PK
prod#	FK to **Product**
cust#	FK to **Customer**
colour	
date_sold	

Product	
prod#	PK
pname	
cost	
maker	FK to **Company**

Avail_Colours	
prod#	PK, FK to **Product**
colour	PK

Figure 4: Colours Schema.

Schema for the Philatelist (Stamp Collecting) Database.

Country(country)
House(title, address, country∗)
  FK (country) refs **Country** — *within*
Series(series#, country, year, currency, denomination)
  FK (country) refs **Country** — *from*
Condition(condition)
Stamp(id, series#, country, year, title∗, condition∗, price)
  FK (series, country, year) refs **Series** — *of*
  FK (title) refs **House** — *owns*
  FK (condition) refs **Condition** — *in*
Listing(series#, country, year, condition, price)
  FK (series, country, year) refs **Series** — *for*
  FK (condition) refs **Condition** — *at*
Assessor(title, emp#, name∗, since∗)
  FK (title) refs **House** — *works*
Assessment(title, emp#, id, series#, country, year, when, price, description)
  FK (title, emp#) refs **Assessor** — *who*
  FK (id, series, country, year) refs **Stamp** — *what*

Figure 5: Philatelist Schema.

Reference

E/R diagram hand-drawing guide.

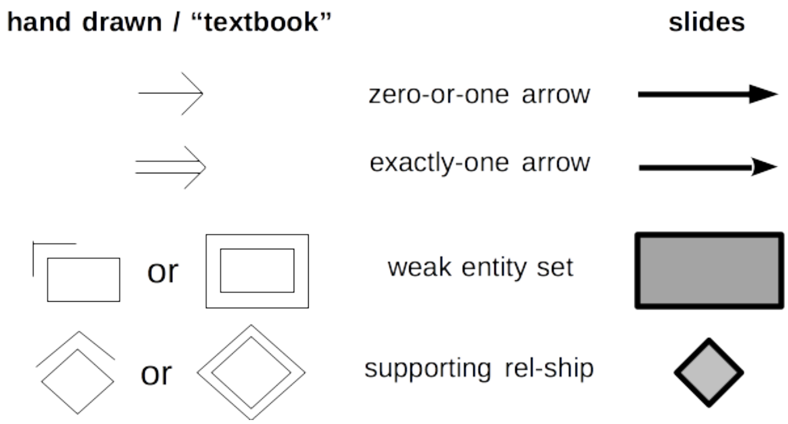


Figure 6: E/R drawing guide.

The Normal Form Definitions.

1NF: Domain of each attribute is an *elementary* type; that is, not a *set* or a *record structure*.

2NF: Whenever $\mathcal{X} \mapsto A$ is a functional dependency that holds in relation **R** and $A \notin \mathcal{X}$, then either

- A is *prime*, or
- $\mathcal{X}$ is not a proper subset of any key for **R**.

3NF: Whenever $\mathcal{X} \mapsto A$ is a functional dependency that holds in relation **R** and $A \notin \mathcal{X}$, then either

- A is *prime*, or
- $\mathcal{X}$ is a key or a super-key for **R**.

BCNF: Whenever $\mathcal{X} \mapsto A$ is a functional dependency that holds in relation **R** and $A \notin \mathcal{X}$, then

- $\mathcal{X}$ is a key or a super-key for **R**.

An attribute A is called *prime* if A is in any of the candidate keys.

Figure 7: The Normal Forms.