COMP 3005 Fall 2012 Mid-Term Test

(Write on the test paper — use the back if necessary;)

Name: __ANSWERS Total 40 Marks_   Student Number:___________

The following definitions are to be used for answering any related questions on the test. These definitions are consistent with those in your notes. Notice that the definition for prime attribute refers to **any** key, not just the primary key of a table schema.

Prime Attribute: If R=(A1,A2,…An) is a table, attribute Ai is prime if there exists a key K of R such that Ai is an element of K. If an attribute is not **prime** it is called **non-prime**.

Trivial Dependency: A dependency X->Y is **trivial** if Y is a subset of X.

Closure of F: if F is a set of functional dependencies, the closure of F, denoted F^+, is {X->Y| F logically implies X->Y}.

Partial Dependency: Suppose X is a key of table R and Y is a proper subset of X, and A is an attribute not in Y. Then Y->A is a **partial dependency**.

2nd Normal Form: A table R with associated functional dependencies F is in 2nd normal form if F^+ contains no partial dependencies Y->A where A is non-prime.

Transitive Dependency: Let Y be a set of attributes from table R and A be an attribute not contained in Y. The functional dependency Y->A is a **transitive dependency** if Y is neither a superkey of R nor a proper subset of a key of R.

3rd Normal Form: A table, with dependencies F, is in 3rd normal form if it is in 2nd normal form and if F^+ contains no transitive dependencies Y->A where A is non-prime. (Equivalently, a table is in 3rd normal form if, for each non-trivial dependency Y->A, Y is a superkey or A is prime.)

Boyce-Codd Normal Form: A table, with dependencies F, is in BCNF if F^+ contains no partial or transitive dependencies. (Equivalently, a table is in BCNF if the left side of each non-trivial dependency in F^+ is a superkey.)

Functional Dependency Inference Rules:

if Y is a subset of X then X->Y	//reflexive rule
if X->AB then X->A and X->B	//decomposition rule
if X->Y and Y->Z then X->Z	//transitive rule
if X->A, and X->B, then X->AB	//union rule
if X->Y then WX->WY	//augmentation rule
if X->Y and WY->Z then WX->Z	//pseudo transitive rule

Question 1 [12 marks]

For each question below a relation R has been defined over attributes U,V,W,X,Y,Z along with a set of functional dependencies that apply to them. In each case provide the possible candidate keys for the relation, state the highest normal form the table R=UVWXYZ would currently satisfy, then show the decomposition of R into as few tables as possible such that each resulting tables are in 3rd Normal Form. Show the primary key for each table in the decomposition by underlining it.

1a) F={UV->WXYZ, X->W, V->X, }

Candidate Keys: UV [1 mark]
Current Normal Form: = 1nf //V->X is partial that violates 2nf [1mark]

3NF Decomposition: <u>UV</u>WYZ <u>V</u>X [1 mark]

1b) F={ UV-> WXYZ, VW->X, }

Candidate Keys: UV [1 mark]

Current Normal Form = 2nd NF [1 mark]

3NF Decomposition: <u>UV</u>WYZ <u>VW</u>X [1 mark]

1c) F={ UV-> WXYZ, X->V, YZ->V}

Candidate Keys: UV, UYZ, UX [1 mark]
Current Normal Form: = 3nf [1mark]

3NF Decomposition: <u>none required</u> [1 mark]

1d) F={ UV-> WXYZ, U->V }

Possible Keys: U [1 mark]
Current Normal Form = 3nf [1mark]

3NF Decomposition: <u>U</u>VXYZ //no decomposition required [1 mark]

Question 2[10 marks]

A rock and roll band wishes to create a database to keep track of their performances (gigs) at local bars, arena's, corporate parties etc. The database will be used to print out set lists for each performance and to generate invoices to the venues so that they can be paid for the performance. The CRTC (Canadian Radio-Television and Telecommunications Commission) also requires that "Cue Sheets" be submitted for any public performance which exceeds 500 spectators. A cue sheet lists the song, author and publisher and the date and location where the song was performed. Cue sheets are used to ensure that the author and publisher get royalty payments.

Each performance (or gig) happens on a particular date, and time, at a particular venue (address). The database should also keep track of the contact person at the venue, to whom the invoice will be sent. Each gig will also have an expected and actual audience size. A performance consists of a number of sets (typically three or four). During each set a number of songs are played in a particular order. A typical set is 45 minutes long; between sets the band takes a 15 minute break. The database will be used to print out set lists for each musician for each gig. A set list simply lists the songs to be performed in that set and the sequence in which they will be played. Each performance can have different set lists.

The database should also record for each song that the band performs, what instruments are needed in that song. This will allow the band manager to make sure that all the proper instruments are sent to the venue for that performance.

Based on an analysis of the above here is a partial list of all the attributes that the database designer thinks needs to be stored in the database. The comments indicate what the attribute represents. Note this list is missing a few attributes that you will have to add in the questions that follow.

```
pid       //unique ID for performance
date      //performance date
time      //performance time
address   //performance venue address
contact   //name of contact person for venue
expected  //expected attendees at performance
actual     //actual attendees at performance
title      //title of a song
author     //author of song
publisher //publisher of a song
sid        // unique id for song
instrument //name of an instrument
id         //unique id for an instrument
```

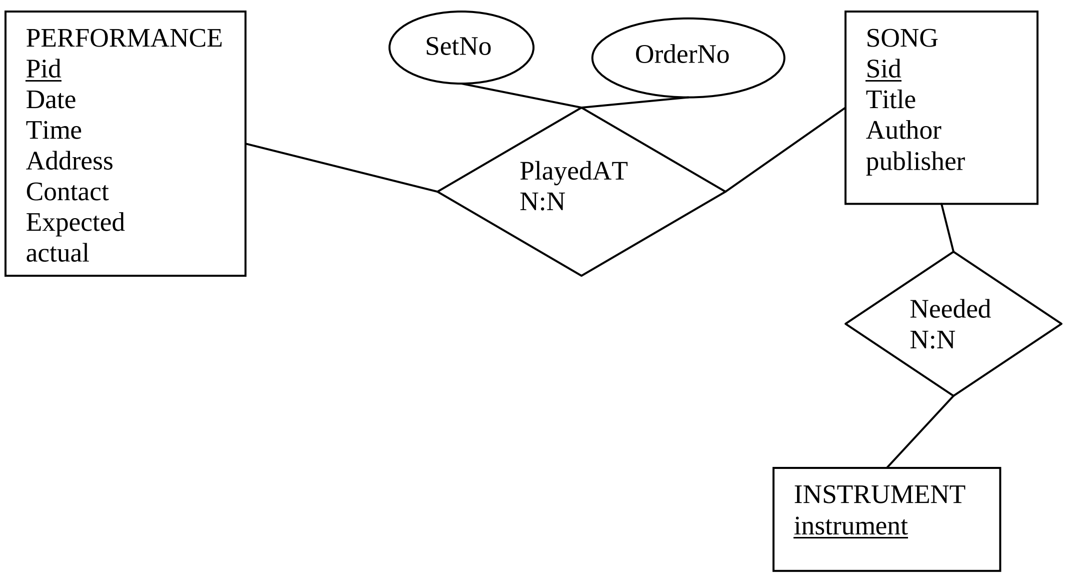
2a) [4 marks] This list of attributes provided on the previous page does not appear to account for the requirements that at a performance songs are performed in sets, and that within a set songs are performed in a particular order. What additional attributes would you add to the list to account for these requirements. (Give the name each attribute and a comment about what it represents.)

```
setNo    //number of a set at a performance
orderNo  //the order of a song in a set
```

1 mark each for attribute and comment. Deduct 1 mark for unnecessary attributes added.

2b) [6 marks] Using the attributes described above and the ones you have added draw an ER diagram for the intended database. Make sure to show the primary key for all entities by underlining them. Also show relationship cardinalities and mandatory participation and any entities that are weak

[-1 mark for any wrong, missing, or unnecessary feature]



Question 3) [8 marks] Consider the set of attributes R={A,B,C,D,E,F} and the following set of functional dependencies proposed by the table designer.

Fd= {ABD->AC, B->E, BA->E, C->BE, AD->FB, C->E }

A colleague suggests that they use the following dependency set instead

Fm = { AD->CF, C->B, B->E}

To do this we would have to prove that the two sets are equivalent.

3a)[4 marks] Using Armstrong's axioms or the six inference rules of functional dependencies, show that the dependency AD->CF in Fm is logically implied by those in Fd (Show clearly each time one of the rules is applied)

AD>-CF implied because :

AD->BF, given in Fd
AD->F decomposition rule

ABD-AC given in Fd
ABD->C decompostion rule

AD->B given in in Fd
AD->ABD augmentation rule
AD- >C transitive rule : AD- >ABD, ABD->C

AD->CF union rule, AD->C, AD->F

3b)[4 marks] (Now the reverse) Using Armstrong's axioms or the six inference rules of functional dependencies, show that the dependency ABD->AC in Fd is logically implied by those in Fm (Show clearly each time one of the rules is applied)

ABD>-AC implied because :

AD->CF, given in Fm
AD- >ACF augmentation rule (add A to both sides)
ABD->ABCF augmentation rule (add B to both sides)
ABD- >AC decomposition rules (ABD->AC, ABD->BF)

Question 4) [10 marks] The following tables have been implemented to represent the telephone switching example described in the course and your assignments. The keys are underlined in the table descriptions. Important foreign keys are shown as // comments in the table descriptions.

```
TABLES:
CALLS //describes calls in progress
FACILITIES //represents lines trunks or treatments by portid
LINES //represents lines of the switch including their directory number
SERVICES  //services customer lines can subscribe to
SERVICE_SUBSCRIBERS  //services that lines currently subscribe to
SUBSCRIBERS //customers who rent lines
TREATMENTS //treatments that can be applied to lines if a call fails
TRUNKS  //trunks that connect to other (foreign) switches
TRUNK_CHANNELS //use of trunk channels
TRUNK_ROUTES  //area codes and office codes served by trunks

facilities
 Name                               Null?    Type
 -------------------------------- -------- ----
 PORTID                             NOT NULL NUMBER(38)
 FACILITY_TYPE                               VARCHAR2(10)

lines
 Name                               Null?    Type
 -------------------------------- -------- ----
 PORTID                             NOT NULL NUMBER(38)
 AREA_CODE                                   CHAR(3)
 OFFICE_CODE                                 CHAR(3)
 STATION_CODE                                CHAR(4)
 STATE                                       CHAR(4) //"busy" or "idle"

trunks
 Name                               Null?    Type
 -------------------------------- -------- ----
 PORTID                             NOT NULL NUMBER(38)
 END_OFFICE                                  VARCHAR2(10)

services
 Name                               Null?    Type
 -------------------------------- -------- ----
 SERVICE_CODE                       NOT NULL CHAR(3)
 SERVICE_NAME                                VARCHAR2(15)

service_subscribers
 Name                               Null?    Type
 -------------------------------- -------- ----
 LINE_PORT_ID                       NOT NULL NUMBER(38) //references lines
 SERVICE_CODE                            NOT NULL CHAR(3) //ref. services

treatments
 Name                               Null?    Type
 -------------------------------- -------- ----
 TREATMENT_CODE                     NOT NULL CHAR(4)
 PORTID                             NOT NULL NUMBER(38)
 TREATMENT_NAME                              VARCHAR2(15)

trunk_channels
 Name                               Null?    Type
 -------------------------------- -------- ----
 PORTID                             NOT NULL NUMBER(38)
 CHANNEL                            NOT NULL NUMBER(38)
 STATE                                       CHAR(4) //"busy" or "idle"
```

subscribers

```
 Name                            Null?    Type
 ------------------------------- -------- ----
 NAME                            NOT NULL VARCHAR2(25)
 ADDRESS                         NOT NULL VARCHAR2(30)
 PORTID                                   NUMBER(38) //references lines
```

calls

```
 Name                            Null?    Type
 ------------------------------- -------- ----
 CALL_ID                         NOT NULL NUMBER(38)
 ORIG_ID                                  NUMBER(38) //ref. facility
 ORIG_CHANNEL                             NUMBER(38) // 0 for lines
 TERM_ID                                  NUMBER(38) //ref. facility
 TERM_CHANNEL                             NUMBER(38) // 0 for lines
 DIALED_AREA                              CHAR(3)
 DIALED_OFFICE                            CHAR(3)
 DIALED_STATION                           CHAR(4)
```

trunk_routes

```
 Name                            Null?    Type
 ------------------------------- -------- ----
 PORTID                          NOT NULL NUMBER(38) //ref. trunks
 AREA_CODE                       NOT NULL CHAR(3)
 OFFICE_CODE                     NOT NULL CHAR(3)
```

Example trunk_routes data

```
SQL> select * from trunk_routes;

   PORTID   AREA_CODE   OFFICE_CODE
------------------------------------
      100   613         235
      100   613         232
      101   613         000
      101   416         765
      101   416         763
      101   905         555
      101   905         543
      101   416         238
      102   905         000
      102   416         000
      103   802         000
      103   514         000
      103   000         000
```

For the following questions provide a relational algebra query that would produce a relation with the required information

4a) [3 marks] Write a relational algebra expression that will give the name and address of all the subscribers that are the originator of a call.

PROJ[name,address](subscribers JOIN[port_ID=Orig_PID] calls)

3marks

1 for project, 1 for joining the correct tables, 1 for correct join condition

Their answers might look different but should be logically equivalent.

4b) [3 marks] Write a relational algebra expression that will list the name and address of all subscribers and the number of services to which they subscribe.

[name,adress] F[count *] (Subscribers JOIN[line_port_id = portid] Service_Subscribers)

[1mark] --------[1mark]-----[1marks]

4c) [4 marks] We want to find all available trunk channels to use for a call going to Toronto (area code 416 or 905). Write a relational algebra query that will list the port ID and channel number of **all** the idle channels that can be used to route the call to Toronto.

PROJ[portID, channel](
SEL[state="idle AND (area_code = 416 OR area_code = 905 OR area_code =000)]
(TrunkChannel * TrunkRoutes)
)

[4 marks]
1 mark for recognizing 416, 905, 000 as valid routing area codes
1 mark for selecting only idle channels
1 mark for correct logic (AND and OR etc
1 mark for correct Join (i.e. TrunkChannels with TrunkRoutes and projecting the correct information