COMP 3005
Assignment #2
Due: Oct. 10 @11:59PM

Instruction

1. Do the assignments independently. Copying is not allowed.
2. The database for this assignment is the same as in Assignment #1. Do this assignment directly on this document and rename it with your last + first name and submit to **brightspace**. Scanned handwritten documents *won't* be accepted. Make sure your uploaded file can be opened.
3. You need to download and install Oracle VM version 3 on your personal computer running intel chips in order to run TRC and DRC. Note that they only work partially.

Part 1 Concepts (20 marks)

Explain the following concepts based on the definitions given in the lecture notes. Different answers found online will be marked wrong. The explanation should be complete; i.e, it does not contain any concept not explained here. Each concept is 2 marks.

1. Atomic Value
   An atomic value is a value that cannot be divided.
2. Tuple
   A tuple is a record that is grouped into files (tables in this case). It represents a row in a table.
3. Mini World
   A mini world is some part of the real world that is stored in the database.
4. Database
   A database is an oracle of stored data that is represented as relations that relate to each other.
5. Database System
   A database systems is a database that is created using a DBMS where information is stored in tables.
6. DBA
   A database administrator (DBA) is a someone who is responsible for the management and maintainability of a database. This includes it's design, configuration, access control, the software and hardware tools needed, and more.
7. End User
   In the context of databases, the end user is someone who interacts with the database from an outsider point of view. They query and view the database, but they do not know how the database is designed or structured.
8. Data Model
   A data model is a structure of how data is laid, related and used. For databases, they provide a clear way of how data elements relate to each other and used to design them.
9. Relational Data Model
   Relational data models use tuples to represent data from the real world and the relationships between them. Each relation is represented by a table and the tables are made up of rows (the tuples) and columns.
10. Database Schema
    A database schema is the structure specifications of the data stored in a database management system. It essentially is the blueprint of the database.

Part 2 (80 marks)

Given the employees and projects databases the same as in Assignment #1. Use both Tuple Relational Calculus (TRC) and Domain Relational Calculus (DRC) to express the same queries as in Assignment Submit your query expressions for each query as well as the query result. Each query is 8 marks, 4 for TRC and 4 for DRC. (80)

Employees

E#	Name	Age	Manager
E1	Adams	50	
E2	Blake	40	E1
E3	Clark	35	E1
E4	David	30	E3
E5	Emily	25	E4
E6	Last	20	E5

Workon

E#	P#	Hours
E1	P1	700
E2	P1	300
E2	P2	200
E3	P1	100
E3	P2	200
E3	P3	300
E4	P1	100
E4	P2	200
E4	P3	300
E6	P1	200
E6	P2	300
E6	P3	400
E6	P4	500

Projects

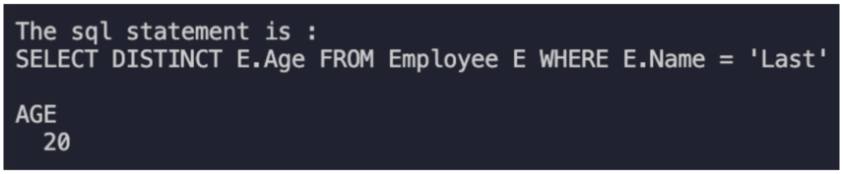
P#	Name	Location
P1	CPU	B1
P2	GPU	B2
P3	GPU	B2
P4	SSD	B3

1) Get the age of Last.

TRC Query:

{E.Age | E in Employee and E.Name = 'Last'};

TRC Result:



```
The sql statement is :
SELECT DISTINCT E.Age FROM Employee E WHERE E.Name = 'Last'

AGE
  20
```

DRC Query:

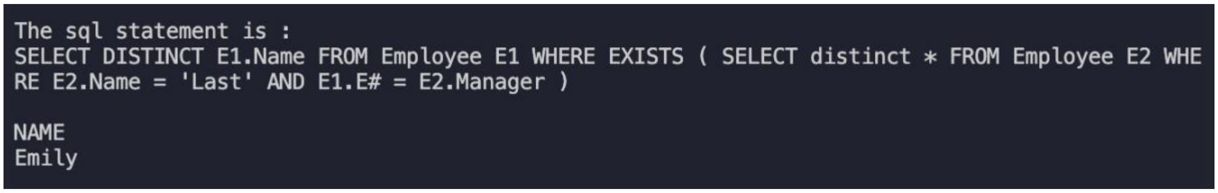
{A | (exists E, M) (Employee (E, 'Last'. A, M)};

2) Get the name of Last's manager

TRC Query:

{E1.Name | E1 in Employee and (exists E2 in Employee)(E2.Name = 'Last' and E1.E# = E2.Manager)};

TRC Result:



```
The sql statement is :
SELECT DISTINCT E1.Name FROM Employee E1 WHERE EXISTS ( SELECT distinct * FROM Employee E2 WHE
RE E2.Name = 'Last' AND E1.E# = E2.Manager )

NAME
Emily
```

DRC Query:

{N | (exists E1,E2,M1, M2) (Employee(E1, 'Last', _, M1) and Employee(E2, N, _, M2) and E2 = M1)};

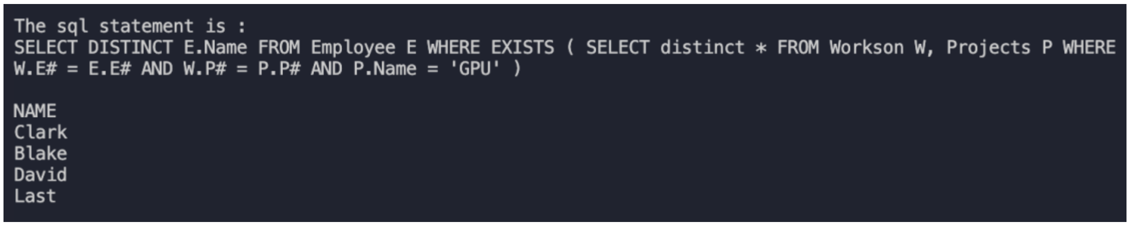
3) Get the name of the employee who works on GPU project.

TRC Query:

{E.Name | E in Employee and

(exists W in Workson, P in Projects)(W.E# = E.E# and W.P# = P.P# and P.Name = 'GPU')};

TRC Result:



```
The sql statement is :
SELECT DISTINCT E.Name FROM Employee E WHERE EXISTS ( SELECT distinct * FROM Workson W, Projects P WHERE
W.E# = E.E# AND W.P# = P.P# AND P.Name = 'GPU' )

NAME
Clark
Blake
David
Last
```

DRC Query:

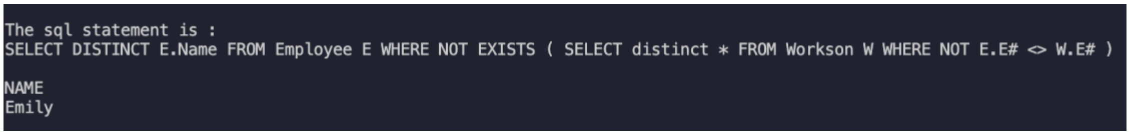
{N | (exists E, P)(Employee(E, N, _, _) and (Workson(E, P, _) and (Projects(P, 'GPU', _))))};

4) Get the name of the employee who does not work on any project.

TRC Query:

{E.Name | E in Employee and (forall W in Workson)(E.E# <> W.E#)};

TRC Result:



```
The sql statement is :
SELECT DISTINCT E.Name FROM Employee E WHERE NOT EXISTS ( SELECT distinct * FROM Workson W WHERE NOT E.E# <> W.E# )

NAME
Emily
```

DRC Query:

{N | (exists E)(Employee(E, N, _, _) and not (exists P)(Projects(P, _, _) and Workson(E, P, _))};

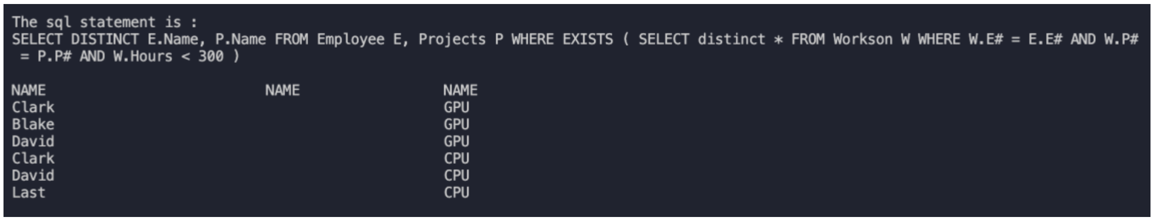
5) Get the pair of employee name and project name such that the employee works on the project less than 300 hours.

TRC Query:

{E.Name, P.Name | E in Employee and P in Projects and

(exists W in Workson)(W.E# = E.E# and W.P# = P.P# and W.Hours < 300)};

TRC Result:



```
The sql statement is :
SELECT DISTINCT E.Name, P.Name FROM Employee E, Projects P WHERE EXISTS ( SELECT distinct * FROM Workson W WHERE W.E# = E.E# AND W.P#
 = P.P# AND W.Hours < 300 )

NAME                          NAME                 NAME
Clark                                              GPU
Blake                                              GPU
David                                              GPU
Clark                                              CPU
David                                              CPU
Last                                               CPU
```

DRC Query:

{EN, PN | (exists E, P, Hours)(Employee(E, EN, _, _) and

Workson(E, P, Hours) and Projects(P, PN, _) and Hours < 300)};

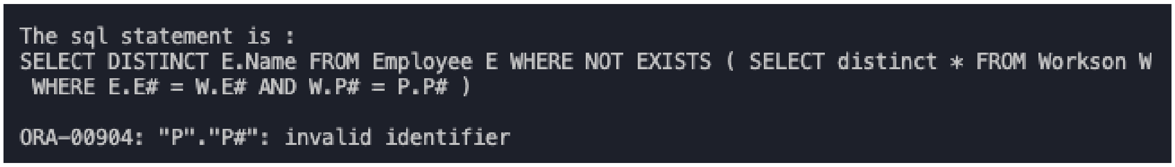
6) Get the name of the employee who works on every project

TRC Query:

{E.Name | E in Employee and

(forall P in Projects) (exists W in Workson)(E.E# = W.E# and W.P# = P.P#)};

Result: error I'm not sure why



```
The sql statement is :
SELECT DISTINCT E.Name FROM Employee E WHERE NOT EXISTS ( SELECT distinct * FROM Workson W
 WHERE E.E# = W.E# AND W.P# = P.P# )

ORA-00904: "P"."P#": invalid identifier
```

The result should have been:

NAME

Last

DRC Query: Taking out the not here, the query would return employees who work on at least one project

{N | (exists E) (Employee(E, N, _, _) and (forall P)(not Projects(P, _, _) or Workson(E, P, _)))};

7) Get the name of the employee who works on every project except SSD.

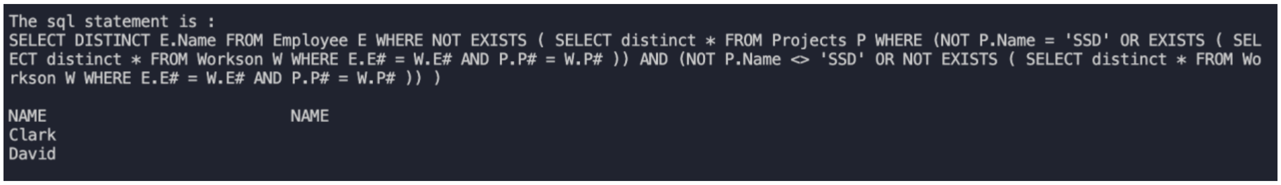
TRC Query:

{E.Name | E in Employee and (forall P in Projects) (

(P.Name = 'SSD' and not(exists W in Workson)(E.E# = W.E# and P.P# = W.P#))

or (P.Name <> 'SSD' and (exists W in Workson)(E.E# = W.E# and P.P# = W.P#)))};

TRC Result:



```
The sql statement is :
SELECT DISTINCT E.Name FROM Employee E WHERE NOT EXISTS ( SELECT distinct * FROM Projects P WHERE (NOT P.Name = 'SSD' OR EXISTS ( SEL
ECT distinct * FROM Workson W WHERE E.E# = W.E# AND P.P# = W.P# )) AND (NOT P.Name <> 'SSD' OR NOT EXISTS ( SELECT distinct * FROM Wo
rkson W WHERE E.E# = W.E# AND P.P# = W.P# )) )

NAME                           NAME
Clark
David
```

DRC Query:

{N | (exists E) (Employee(E, N, _, _) and

(forall P)(not (exists PN)(Projects(P, PN, _) and PN != 'SSD') or Workson(E, P, _))

and (not (exists PN)(Projects(P, PN, _) and PN = 'SSD') or not Workson(E, P)))};

8) Get the name of the employee who works on every project that Clark works on.

TRC Query:

The or not ensures that there are no additional projects that E1 works on that Clark does not work on in the same project scope “P’

{E1.Name | E1 in Employee and E1.Name <> 'Clark' and

(exists E in Employee)(E.Name = 'Clark' and (forall P in Projects)(

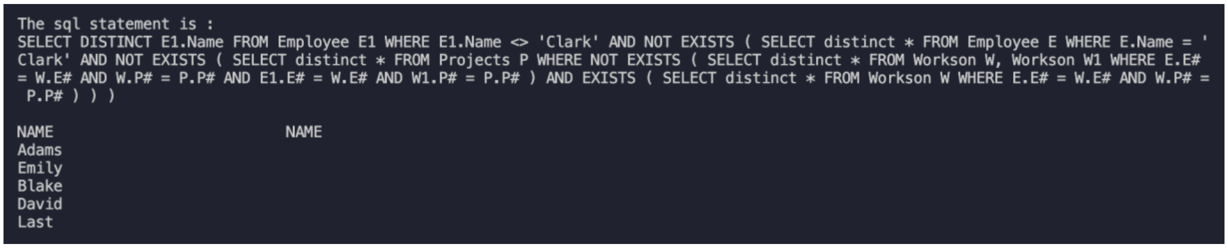
(exists W in Workson, W1 in Workson)

(E.E# = W.E# and W.P# = P.P# and E1.E# = W.E# and W1.P# = P.P#) or

not(exists W in Workson)(E.E# = W.E# and W.P# = P.P#)))};

TRC Result:

not condition checks if E1 works on all of Clark’s projects



```
The sql statement is :
SELECT DISTINCT E1.Name FROM Employee E1 WHERE E1.Name <> 'Clark' AND NOT EXISTS ( SELECT distinct * FROM Employee E WHERE E.Name = '
Clark' AND NOT EXISTS ( SELECT distinct * FROM Projects P WHERE NOT EXISTS ( SELECT distinct * FROM Workson W, Workson W1 WHERE E.E#
= W.E# AND W.P# = P.P# AND E1.E# = W.E# AND W1.P# = P.P# ) AND EXISTS ( SELECT distinct * FROM Workson W WHERE E.E# = W.E# AND W.P# =
 P.P# ) ) )

NAME                           NAME
Adams
Emily
Blake
David
Last
```

DRC Query:

{N | (exists E1, E)(Employee(E1, N, _, _) and N != 'Clark' and

Employee(E, 'Clark', _, _) and (forall P) (not Workson(E, P, _) or Workson(E1, P, _)))};

9) Get the name of the employee who works on the same projects that Clark works on.

TRC Query:

The or not here checks for the absence of projects worked on by E1 and Clark but does not take into account the specific project P being iterated. It checks for the absence of any additional projects where E1 works but Clark does not

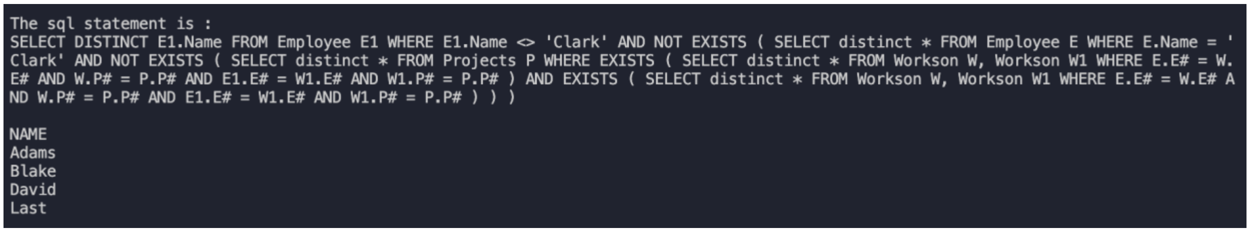
{E1.Name | E1 in Employee and E1.Name <> 'Clark' and

(exists E in Employee)(E.Name = 'Clark' and (forall P in Projects)(

(exists W in Workson, W1 in Workson)(E.E# = W.E# and W.P# = P.P# and E1.E# = W1.E# and W1.P# = P.P#) or

not(exists W in Workson, W1 in Workson)(E.E# = W.E# and W.P# = P.P# and E1.E# = W1.E# and W1.P# = P.P#)))};

TRC Result:



```
The sql statement is :
SELECT DISTINCT E1.Name FROM Employee E1 WHERE E1.Name <> 'Clark' AND NOT EXISTS ( SELECT distinct * FROM Employee E WHERE E.Name = '
Clark' AND NOT EXISTS ( SELECT distinct * FROM Projects P WHERE EXISTS ( SELECT distinct * FROM Workson W, Workson W1 WHERE E.E# = W.
E# AND W.P# = P.P# AND E1.E# = W1.E# AND W1.P# = P.P# ) AND EXISTS ( SELECT distinct * FROM Workson W, Workson W1 WHERE E.E# = W.E# A
ND W.P# = P.P# AND E1.E# = W1.E# AND W1.P# = P.P# ) ) )

NAME
Adams
Blake
David
Last
```

DRC Query:

{N | (exists E1, E)(Employee(E1, N, _, _) and N != 'Clark' and

Employee(E, 'Clark', _, _) and (forall P)

(not Workson(E, P, _) or Workson(E1, P, _)) and

(Workson(E, P, _) or not Workson(E1, P, _)))};

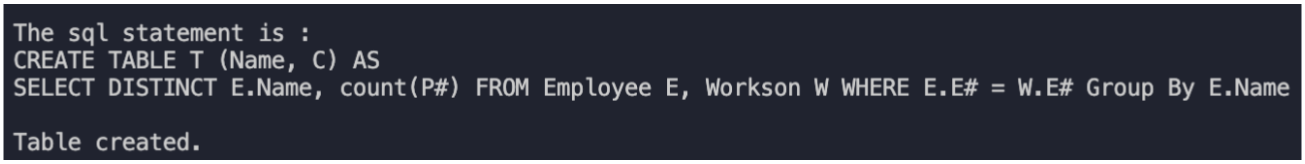
10) Get the name of the employee who works on more than two projects.

TRC Query:

T(Name, C) := {E.Name, count(W.P#) | E in Employee and W in Workson and E.E# = W.E#};

{E.Name | E in T and E.C > 2};

TRC Result:



```
The sql statement is :
CREATE TABLE T (Name, C) AS
SELECT DISTINCT E.Name, count(P#) FROM Employee E, Workson W WHERE E.E# = W.E# Group By E.Name

Table created.
```

DRC Query:

T(Name, C) := {N, count(*) | (exists E) (Employee(E, N, _, _) and Workson(E, _, _))};

{N | (exists C)(T(N, C) and C > 2)};