Technion – Israel Institute of Technology Computer Science Faculty

Lecturer: Prof Benny Kimelfeld

Teaching Assistants: Shoval Lagziel   
 Dvir Dukhan  
 Liat Peterfreund

Database Systmes

236363

18 of July, 2017

Exam Parts and Score

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Score | Number Of Questions you Should answer | Questions in Part | Subject | Part |
| 30 | 2 | 2 | Database Design | 1 |
| 34 | 2 | 2 | Relational Queries | 2 |
| 36 | 3 | 4 | noSQL | 3 |

**Guidelines**

1. You should write your answers on this form and in specified places.
2. You can use any material written on paper.
3. Do not accept or transfer any reference material during the exam.
4. You have a selection between two questions: You must answer one question from Neo4j or MongoDB, explicitly indicate what you chose, in case of unclear markup, the first answer will be checked.
5. Use only marks or functions learned in tutorials or lectures, the use of a non-such marking requires a full explanation of the meaning of the markup.
6. The duration of the exam is three hours. Plan your time accordingly.
7. Eight questions in three parts. Please make sure you have the entire form.

**Good Luck!**

**Part 1 – Database Design – 30 pts - All questions in this section are mandatory**

1. **ERD, 10 pts**

The following is part of the ERD diagram of an auctions system:

### Customer

#### Bid

### Auction

Containss

Item

**Diagram Explanations:**

* **Customer** Entity represents a customer. Cid – customer id, Cname – customer’s name, CCard – Customer’s credit card, CAddr - Customer's address.
* **Auction** Entity represents an auction . AId - Auction ID number, Type - Auction type or Group sale, Closed - Whether the auction is closed or not (Boolean value), CurPrice - The current price (minimum bid).
* **Item** Entity represents an item that is offered for auction. IId - ID number of item, IDesс - textual description of item, Cost - item cost to supply, Quantity - quantity of items in stock.
* **Bid** Relationship describes the submission of a bid by a client in the auction. Price - the proposed price, NumPayments - the number of payments, Win - the offer is credited (Boolean).
* **Contains** Relationship describes the relevance of items to auctions.

1. (3 pts) Fill the following based on the ERD diagram.

Bid Table:

|  |  |  |
| --- | --- | --- |
|  | | Properties |
|  | Key | |

Auction Table:

|  |  |  |
| --- | --- | --- |
|  | | Properties |
|  | Key | |

Contains Table:

|  |  |  |
| --- | --- | --- |
|  | | Properties |
|  | Key | |

b. Is it possible for a customer to have more than one credit card? More than one address? Suggest a change to the ERD so that the credit card answer would be the opposite and that the answer about the address would remain unchanged. Draw the proposed change.

Yes/No Explanation:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

c. (2 points) Can an item be offered simultaneously in more than one auction? Suggest a change to ERD so that the answer would be the opposite.

Yes/No Explanation:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Change:

d. (3 pts) What are the problems that may arise from turning the customer into a weak entity of Auction (through Bid)?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Functional Dependencies and normal forms, 20 pts**

Schema R (A, B, C, D, E, H) is provided and a functional dependencies set F:

a. (6 points) Specify three (minimal) keys for R. (no need to explain)

\_\_\_\_\_\_\_

\_\_\_\_\_\_\_

\_\_\_\_\_\_\_

b. (5 points) Find a minimal cover for F. (no need to explain or show the calculations)

c. (9 pts) The following algorithm is written by the student Adi who is studying in the Database Systems course, which aims to examine whether Schema R with functional dependencies set is in the normal form 3NF:

Note that this section talks about any scheme and any FD set and is not necessarily related to previous sections.

Find a (minimal) key Z of R

For each **nontrivial** FD X ->Y in F {

If X is **not** a superkey then {

For each A in Y\X {

If A is not in Z then return **false**

}

}

}   
Return **true**

1. If the algorithm returns **true**, is Schema R necessarily in the 3NF?

**Yes/No,** Explanation**:**

1. If the algorithm returns **false**, does Schema R necessarily not in 3NF?

**Yes/No,** Explanation**:**

1. Suggest a small change to the algorithm that will make it correct. Explain whether the algorithm after the change is necessarily running at a polynomial time of R and F.

**The suggested algorithm change:**

**Time Complexity explanation:**

**Part 2 - Relational queries - 34 points - All the questions in this part are mandatory**

1. **SQL, RA, 20 pts**

The Babazon shopping site manages their database representing customers, products and purchases by the following tables:

**CUST (Costumers Table)**

|  |  |  |
| --- | --- | --- |
| Cust\_ID | Name | Age |

**PROD (Products Table)**

|  |  |
| --- | --- |
| Prod\_ID | Price |

**PURCH (Purchases Table)**

|  |  |
| --- | --- |
| Cust\_ID | Prod\_ID |

1. Write SQL queries that return the average prices of the products each customer has purchased for customers who have made more than 5 purchases and their customer ID.

In this section you must write two queries in different forms as described below, which return the same output.

1. (4 pts) Write a SQL query that returns the output without ‘HAVING’ (nested queries can be used)

1. (4 pts) Write a SQL query that returns the output with ‘HAVING’ and does not nest queries.
2. (4 points) Write a SQL query that returns for each product identifier the names of the youngest costumers who bought it. (the query will return pairs of product id and customer id)

1. (2 pts) Write a RA query that returns the names of all the customers who bought all the products in the database.
2. (6 points) Can the division operator of relational algebra be expressed by using projection, selection, Cartesian product, Union, and Rename operators (σ × ∪ ρ)? If so, show the formula that defines division; if not, prove.
3. **DATALOG, RC, 14 pts**

Babazon wants to build a recommendations system which recommends about products similar to the products the customer purchased or interested in, depending on the characteristics of the customer or the characteristics of the product.

In this question, you must assume that the tables presented in the previous question exist.

For customer c who bought product p we define a recommended product p' - a product different from p which purchased by another client c', such that c and c’ are the same age and c’ also bought p.

1. (5 points) Write an RC query that returns for each customer the **recommended products ids and the price.**

You can use the following relations

Note that you must actually return triplets in the following form:  
 (CustId, RecommededProductId, price)

1. (5 pts) The following rating function is used to rank the recommendations: two different users are **similar** if they are the same age, and there is a pathway between them such that all customers are the same age, and each two neighbors in the pathway purchased at least one same product.

Write a Datalog program that produces a Predicate named Similar(id1, id2) so it true for two users id1, id2 if they are **similar**.

1. (4 pts) Consider the two following EDB predicates

Vertex

|  |
| --- |
| Vertex\_id |

Edge

|  |  |
| --- | --- |
| Vertex\_id1 | Vertex\_id2 |

And the following program for finding nodes which are not neighbors in the graph:

Show two stratifications for the program, such that one is legal and is not. Explain.

**Legal Stratification:**

**Explanation:**

**Illegal Stratification:**

**Explanation:**

**Part 3- noSQL – 36 pts – questions 5 and 6 are mandatory, and you have to choose between question 7 and 8.**

**5. XML, 13 pts, a mandatory question**

In this question, we will assume the following DTD declaration.

<!DOCTYPE courses [

<!ELEMENT courses (faculty\*)>

<!ELEMENT faculty (course+)>

<!ATTLIST faculty name CDATA #REQUIRED>

<!ELEMENT course (prereq?,lecturer+)>

<!ATTLIST course number ID #REQUIRED

name CDATA #IMPLIED

grad (yes|no) "no"

>

<!ELEMENT prereq EMPTY>

<!ATTLIST prereq prenum IDREF #REQUIRED>

<!ELEMENT lecturer (#PCDATA)>

]>

a. Answer the following yes/no questions and explain your answer. Both questions apply only to documents that conform to the above DTD.

(1pts) Can there be a course that is a prerequisite of itself?

**Yes/No , Explanation:**

(2pts) Can the attribute prenum hold an identifier of an element that is not a course?

**Yes/No , Explanation:**

b. Write XPath patterns that return the following information from every document that conforms to the courses DTD. Note that you can use only XPath 1.0, that is, the version studied in class (as opposed to, e.g., XPath 2.0).

All faculties that contain at least one graduate course (@grad=”yes”)

**XPath pattern:**

All courses that are prerequisites (@prereq) of courses from their own faculties.

**Xpath pattern:**

c. For each of the following XPath patterns, write down an XML document that conforms to the above DTD and has a nonempty answer for the pattern, or explain why no such document exists. Note that you can reuse an answer document across different patterns.

1. //faculty/course[@grad="yes"][../\*/@grad="no"]
2. //faculty[count(course)>1 and count(course[lecturer])=1]
3. //faculty[count(course)>1 and count(./\*[@name])=0]

**6. RDF, 13pts, a mandatory question**

Consider the following RDF graph.

|  |  |  |
| --- | --- | --- |
| dbr:C1 | dbp:mayor | dbr:M1 |
| dbr:M1 | dbp:birthPlace | dbr:C22 |
| dbr:C22 | dbp:mayor | dbr:M22 |
| dbr:M22 | dbp:birthPlace | dbr:C1 |
| dbr:M22 | **rdf:type** | dbo:person |
| dbr:C333 | dbp:mayor | dbr:M333 |
| dbr:M333 | dbp:birthPlace | dbr:C333 |
| dbo:mayor | **rdfs:range** | dbo:official |
| dbo: official | **rdfs:subClassOf** | dbo:person |

1. (6 pts) For each of the following queries, write down the result under the ordinary (non-RDFS) semantics.

|  |
| --- |
| SELECT ?c, ?p WHERE {  { ?c dbp:mayor ?m. }  **UNION** {  ?p dbo:birthPlace ?c.  ?p rdf:type dbo:person. }  } |

Results:

|  |
| --- |
| SELECT ?c, ?m WHERE {  { ?c dbp:mayor ?m. }  **MINUS** {  ?p dbo:birthPlace ?c.  ?p rdf:type dbo:person. }  } |

Results:

c. שאילתא:

|  |
| --- |
| SELECT ?c ?m WHERE {  { ?c dbo:mayor ?m.  OPTIONAL {  ?p dbp:birthPlace ?c.  ?p rdf:type dbo:person.  }  }  MINUS {  ?c dbp:mayor ?p.  }  } |

Results:

1. (3 pts) Write down the result of the last Pattern from Section a, this time under the RDFS semantics.

Results

1. (4 pts) Explain in words what the following SPARQL query returns:

SELECT DISTINCT ?c ?g ?p WHERE {

?c rdf:type dbo:City.

?c owl:sameAs ?d.

GRAPH ?g { ?d ?p ?o}

}

Explanation:

1. **Neo4j, Select one from 7,8**

Consider the following neo4j database, representing a social network that contains the following type of nodes (for each node single label).

|  |  |  |
| --- | --- | --- |
| **Group** | **Post** | **Person** |
| Name | Title | Name |
|  | Text | ID |

The edges in the database are from the following types: (each edge has one label)

**Likes**: Connects between a Person and a Post  
**Posted**: Connects between Person and Post  
**BelongsTo**: Connects between a Person and a Group  
**FriendsOf**: Connects between a Person and a Person

Suppose that in our database friendship is symmetric and therefore if X is a friend of Y so there will be edges from X to Y and from Y to X.  
*Note: Too long and cumbersome answers may cause a reduction of points.*

1. Write a Cypher query which returns:

The names of all the people that Like **all** posts posted by ‘Michal’.

1. (5 points) We declare **distance between people** as the **minimal** pathway length (number of edges) which contains only edges of type *FriendsOf*.

Write a Cypher query that returns the largest distance between two people belonging to a group called MAMAN. (You can assume that there are at least two people belonging to the MAMAN group whose distance is greater than 0)

**8. MongoDB, 10 pts, select one from 7,8**

Search engines fill in the following task: Given a collection of documents and search keywords, return the most relevant documents to your keywords.  
Search engines use “**inverted index”** as part of the query response process. A basic inverted index will look like the following:

**{**

\_id **:** /\*a token string\*/**,**

Value**:** **{**

Docs**:** **[**

**{**

Doc\_id**:** /\*document id\*/

Appearances**:**/\*number of appearances of

token in the document\*/

**}**

**]**

**}**

**}**

When token is a word and doc\_id is an array that contains the identifiers of the documents that contain the token word.

Under the moogle database, you are given a collection of intermediate documents, which contains an intermediate calculation performed for you (similar to the word count exercise in the homework) that holds for each word and document the number of occurrences of the word in the document.

Each document (in the intermediate collection) consists of the following:

**{**

\_id**:**

**{**

Token **:** /\*a string token\*/

Doc\_id**:** /\*the document id\*/

**}**

value**:** /\*number of appearances of the word in the document\*/

**}**

1. (7 points) You need to create the **moogle\_ii** collection that is an inverted index using map-reduce.

**var map = function()**

**{**

**}**

**var reduce = function (key, values)**

**{**

**}**

**db.intermediate.mapReduce(**

**)**

1. (3 pts) this question is independent on your solution of section a

What does the following query return?

**Pay attention:**

Split (delimiter) splits a string by delimiter into a set of strings.

$In: array Checks whether a given value is in an array.

**var** query **=** "........"//some input query for the system

db**.**moogle\_ii**.**aggregate**(**

**{**

$match **:** **{**\_id **:** **{** $in**:** query**.**split**(**" "**)}}**

**},**

**{**

$unwind**:** '$value.Docs'

**},**

**{**

$group**:** **{**

\_id**:** "$value.Docs.Doc\_id"**,**

Total **:{**$sum **:**"$value.Docs.Appearances" **}**

**}**

**},**

**{**

$sort**:{**Total **:** **-**1**}**

**}**

**)**

**Answer:**

If you are using this page, mark this next to the original question (s), and write the question numbers here. Question: \_\_\_\_\_\_\_\_\_ Section: \_\_\_\_\_\_\_\_

Question: \_\_\_\_\_\_\_\_\_ Section: \_\_\_\_\_\_\_\_

Question: \_\_\_\_\_\_\_\_\_ Section: \_\_\_\_\_\_\_\_

Question: \_\_\_\_\_\_\_\_\_ Section: \_\_\_\_\_\_\_\_