Introduction

Examination, INF3100, 2017.

No examination support material is allowed

In this exam, you can get a maximum of 100 marks. Each problem is worth 5 or 10 marks (shown on the problem). Note that several problems explicitly ask you to explain your answer; it is important that you do so.

Section 4 has two problems that you must solve using digital hand drawing (scantron).
In this exam, you are permitted to make drawings/use sketching for task 4.1 and 4.2. You are to use the sketching paper handed to you in the exam room. You can use more than one sketching sheet per task. See instructions for filling out sketching sheets on your desk.

You may NOT hand in sketching sheets for any other tasks than task 4.1and 4.2. You will NOT be given extra time to fill out the "general information" on the sketching sheets (task codes, candidate number etc.)

Good luck!

2.1 Functional dependency

Consider the relation

Review(Book, Author, Publisher, User, RevText, Rating)

which stores information about book reviews. **RevText** is the review text that the **User** wrote for a particular **Book** by a particular **Author**. The book is published by a **Publisher**, and the **User** also gives the book a **Rating**.

We would like to add the rule that all books by the same author are published by the same publisher. Formulate this rule as a functional dependency (FD). Use "->" for the arrow symbol.

Fill in your answer here

Format

Words: 0

Maximum marks: 5

2.2 Normal forms

Consider the relation

Review(Book, Author, Publisher, User, RevText, Rating)

which stores information about book reviews. **RevText** is the review text that the **User** wrote for a particular **Book** by a particular **Author**. The book is published by a **Publisher**, and the **User** also gives the book a **Rating**.

For this question, assume that the only functional dependencies we have are

User, Book -> RevText
User, Book -> Rating
Book -> Author
Author -> Publisher

What normal form does the relation Review satisfy? Explain your answer.

Fill in your answer here

Format

Words: 0

Maximum marks: 10

2.3 Decomposition

Consider the relation

Review(Book, Author, Publisher, User, RevText, Rating)

which stores information about book reviews. **RevText** is the review text that the **User** wrote for a particular **Book** by a particular **Author**. The book is published by a **Publisher**, and the **User** also gives the book a **Rating**.

For this question, assume that the only functional dependencies we have are

User, Book -> RevText
User, Book -> Rating
Book -> Publisher

Give a lossless decomposition of Review into BCNF. Feel free to invent sensible names for the new relations. Explain your answer.

Fill in your answer here

Format

Words: 0

Maximum marks: 10

3.1 SQL, above average karma

Consider the following simple setup for a blog database.

Post(Id, Title, User, Body, Timestamp, Score)
Comment(Id, User, Body, Timestamp, ReplyTo, RefPost, Score)
User(Id, DisplayName, Karma)

The **Post** table holds entries about blog posts, with an **Id**, a **Title** and a **Body** (the text of the post), a **Timestamp**, and a **Score** (users get to give and take away points).

Likewise, the **Comment** table holds information about comments left on a post. Comments are in reference to a post (**RefPost** holds this information, cannot be NULL), and can be in reply to other comments (**ReplyTo** stores this information, can be NULL).

Users have a karma rating based off the scores on their posts and comments.

Id is the primary key in all three tables.
Comment.ReplyTo is a foreign key referencing **Comment.Id.**
Comment.RefPost is a foreign key referencing **Post.Id.**
Post.User and **Comment.User** are foreign keys referencing **User.Id.**

Write a query to find the users who have more karma than the average for all users. For each such user, print the DisplayName as well as the Id and Score of their highest-scoring posts.

Fill in your answer here

```
1
```

Maximum marks: 5

3.2 SQL, self-replies

Consider the following simple setup for a blog database.

Post(Id, Title, User, Body, Timestamp, Score)
Comment(Id, User, Body, Timestamp, ReplyTo, RefPost, Score)
User(Id, DisplayName, Karma)

The **Post** table holds entries about blog posts, with an **Id**, a **Title** and a **Body** (the text of the post), a **Timestamp**, and a **Score** (users get to give and take away points).

Likewise, the **Comment** table holds information about comments left on a post. Comments are in reference to a post (**RefPost** holds this information, cannot be NULL), and can be in reply to other comments (**ReplyTo** stores this information, can be NULL).

Users have a karma rating based off the scores on their posts and comments.
Id is the primary key in all three tables.
Comment.ReplyTo is a foreign key referencing **Comment.Id.**
Comment.RefPost is a foreign key referencing **Post.Id.**
Post.User and **Comment.User** are foreign keys referencing **User.Id.**

Write a query to count direct self-replies, that is, comments by a user left in reply to their own comments (both the reply and the comment are written by the same user) per user per post. Print the post title, user id, and number of self-replies.

Fill in your answer here

```
1
```

Maximum marks: 10

3.3 Recursive SQL

Consider the following simple setup for a blog database.

Post(Id, Title, User, Body, Timestamp, Score)
Comment(Id, User, Body, Timestamp, ReplyTo, RefPost, Score)
User(Id, DisplayName, Karma)

The **Post** table holds entries about blog posts, with an **Id**, a **Title** and a **Body** (the text of the post), a **Timestamp**, and a **Score** (users get to give and take away points).

Likewise, the **Comment** table holds information about comments left on a post. Comments are in reference to a post (**RefPost** holds this information, cannot be NULL), and can be in reply to other comments (**ReplyTo** stores this information, can be NULL).

Users have a karma rating based off the scores on their posts and comments.
Id is the primary key in all three tables.
Comment.ReplyTo is a foreign key referencing **Comment.Id.**
Comment.RefPost is a foreign key referencing **Post.Id.**
Post.User and **Comment.User** are foreign keys referencing **User.Id.**

Write a recursive query that prints all replies to the comment with ID 3415. That is, all direct replies as well as replies to replies.

Fill in your answer here

```
1
```

Maximum marks: 5

4.1 Rel. alg., multiple reviews

In this task you are to hand in sketches. Use the sketching paper handed to you in the exam room for this. See instructions on your desk.

Consider the relation

Review(Book, Author, Publisher, User, RevText, Rating)

which stores information about book reviews. **RevText** is the review text that the **User** wrote for a particular **Book** by a particular **Author**. The book is published by a **Publisher**, and the **User** also gives the book a **Rating**.

We would like to add the constraint that no **User** may review the same **Book** more than once. Formulate this constraint using relational algebra.

Maximum marks: 5

4.2 Rel. alg., query

In this task you are to hand in sketches. Use the sketching paper handed to you in the exam room for this. See instructions on your desk.

Consider the relation

Review(Book, Author, Publisher, User, RevText, Rating)

which stores information about book reviews. **RevText** is the review text that the **User** wrote for a particular **Book** by a particular **Author**. The book is published by a **Publisher**, and the **User** also gives the book a **Rating**.

Write a query to retrieve pairs of **distinct** users that gave the same book the same rating as **a relational algebra expression**. Do not worry about the same pairs appearing reversed.

Maximum marks: 10

5.1 Cascading rollback

Explain the concept of cascading rollback in transaction management.

Fill in your answer here

Format | B I U x_2 x^2 I_x | ABC

Words: 0

Maximum marks: 5

5.2 Conflict serializability

Consider the transaction schedule

$r_1(A); w_1(B); r_2(B); w_2(C); r_3(C); w_3(A);$

where $r_1(X)$ means that transaction 1 reads element X, and $w_1(X)$ that transaction 1 writes X.

Is this schedule conflict serializable? Explain your answer.

Fill in your answer here

Format | B I U x_2 x^2 | I_x | ✂ ⧉ 📋 | ↩ ↪ ↻ | 1. • | Ω ▦ ✎ | Σ | ABC | ⛶

Words: 0

Maximum marks: 5

5.3 First updater wins

Consider the following transaction schedule.

$r_1(A)$; $w_2(A)$; $w_3(B)$; $w_1(B)$; $w_3(A)$;

where $r_1(X)$ means that transaction 1 reads element X, and $w_1(X)$ that transaction 1 writes X.

Assume that the actions start in the order given, for example, $w_3(B)$ is attempted before $w_1(B)$. Describe the execution of these three transactions under snapshot isolation using the first updater wins (FUW) rule. That is, write down who requests and is granted which write locks, who has to wait, who gets to commit and who has to abort, and so on.

Fill in your answer here

Format | B I U x_2 x^2 | I_x | ✂ ⧉ 📋 | ↩ ↪ ↻ | 1. • | Ω ▦ ✎ | Σ | ABC | ⛶

Words: 0

Maximum marks: 10

6.1 2PC, part 1

Assume a Two Phase Commit (2PC) protocol with one Transaction Controller (TC) and two nodes A and B. At the end of first phase, the TC received “yes” (ready to commit) messages from both nodes A and B. In phase two, assume node B contacts node A and discovers that node A has not yet received a commit or abort message from the TC.

Would it be correct for node B to commit or abort at this stage? What about A, should it commit or abort? Explain your answer.

Fill in your answer here

Format

Words: 0

Maximum marks: 5

6.2 2PC, part 2

Assume a Two Phase Commit (2PC) protocol with one Transaction Controller (TC) and two nodes A and B. At the end of first phase, the TC received “yes” (ready to commit) messages from both nodes A and B. In phase two, assume node B contacts node A and discovers that node A has not yet received a commit or abort message from the TC.

Describe a sequence of events (messages sent/received, network partitions, node crashes, etc.) that could lead A to commit but not B.

Fill in your answer here

Format | B I U x_2 x^2 | I_x | ... | Ω | Σ | ABC

Words: 0

Maximum marks: 5

7.1 Estimation

Consider the following relations

A(x, y, z)
B(x, v)
C(w, u)

Using T(R) to denote the number of tuples in relation R, and V(R, A) to denote the number of distinct values for attribute A in relation R, write down an estimate for the number of tuples in the following relation:
$\sigma_{x=625}((A \bowtie B) \bowtie_{v<w} C)$

Fill in your answer here

Format | B I U x_2 x^2 | I_x | ... | Ω | Σ | ABC

Words: 0

Maximum marks: 5

7.2 Query plans

Consider the following relations

A(x, y, z)
B(x, v)
C(w, u)

Consider the two equivalent, logical query plans (that is, brackets define the order in which operations are carried out)

$$\sigma_{x=625}((A \bowtie B) \bowtie_{v<w} C)$$

and

$$(\sigma_{x=625}(A) \bowtie B) \bowtie_{v<w} C$$

Which one is likely to use less memory? Explain your answer.

Fill in your answer here

Format

Words: 0

Maximum marks: 5