Name

CWID

# Quiz
# 1

## Sept 22th, 2020
## Due Sept 30st, 11:59pm

# Quiz 1: CS525 - Advanced Database Organization

---

*Please leave this empty!* 1.1 1.2 1.3 1.4 Sum

# Instructions

- **You have to hand in the assignment using your blackboard**
- **This is an individual and not a group assignment**
- Multiple choice questions are graded in the following way: You get points for correct answers and points subtracted for wrong answers. The minimum points for each questions is **0**. For example, assume there is a multiple choice question with 6 answers - each may be correct or incorrect - and each answer gives 1 point. If you answer 3 questions correct and 3 incorrect you get 0 points. If you answer 4 questions correct and 2 incorrect you get 2 points. ...
- For your convenience the number of points for each part and questions are shown in parenthesis.
- There are 4 parts in this quiz
    1. SQL
    2. Relational Algebra
    3. Index Structures
    4. Result Size Estimation

## Part 1.1 SQL (Total: 31 + 10 bonus points Points)

Consider the following publication schema and example instance. The example data should not be used to formulate queries. SQL statements that you write should return the correct result for every possible instance of the schema!

### Author

| aname | affiliation |
|---|---|
| Peter Smith | University of Science |
| Anish Kapoor | UNC |
| Yutang Li | UNC |

### AuthorPublication

| aname | title |
|---|---|
| Peter Smith | A Mole of moles |
| Peter Smith | A tractus on logic |
| Anish Kapoor | Analyzing the frequency of studies: a meta review |
| Yutang Li | Analyzing the frequency of studies: a meta review |

### Article

| title | journal | number | issue |
|---|---|---|---|
| A Mole of moles | Journal of P-Hacking | 1 | 12 |
| A tractus on logic | Journal of Bogus Research | 3 | 1 |
| Analyzing the frequency of studies: a meta review | Science | 4 | 3 |

### Journal

| jname | field | pagelimit | publisher |
|---|---|---|---|
| VLDB Journal | CS | 20 | Springer |
| Science | All | 12 | Springer |
| Journal of Bogus Research | All | 8 | Heist |
| Journal of P-Hacking | Statistics | 12 | Heist |

### Publisher

| pname | location | type |
|---|---|---|
| Springer | Berlin | Academic |
| Heist | Bern | Predatory |
| EST | Charlotte | Academic |

**Hints:**

- Attributes with black background are the primary key attributes of a relation
- Attribute `aname` of relation `AuthorPublication` is a foreign keys to relation `Author`.
- Attribute `title` of relation `AuthorPublication` is a foreign keys to relation `Article`.
- Attribute `journal` of relation `Article` is a foreign keys to relation `Journal`.
- Attribute `publisher` of relation `publisher` is a foreign keys to relation `Publisher`.

### Question 1.1.1 (3 Points)

Write an SQL query that returns the names of researchers from the university of science that have exclusively published in predatory journals (`type` is "*Predatory*").

### Question 1.1.2 (5 Points)

Write an SQL query that returns the names of authors that have published in the same journals, i.e., a pair of authors *author1* and *author2* should be return if the set of journals *author1* published in equal to the set of journals *author2* published in.

### Question 1.1.3 (5 Points)

Write a query that returns the journal with the highest growth factor averaged across all of the issues of the journal. The growth factor of issue $x$ of a journal is defined as the number of articles published in this issue divided by the number of articles published in issue $x - 1$ (this is only defined for $x > 1$).

### Question 1.1.4 (2 Points)

Return the names of the 3 researchers that have published the most articles.

### Question 1.1.5 (2 Points)

Write an SQL query that returns publishers that publish more than 5 journals.

### Question 1.1.6 (3 Points)

Write an SQL query that returns pairs of authors $(a_1, a_2)$ that are direct co-author (have published an article together) or indirect co-authors (authors $a_1$ has published with a third author $a_3$ who is a direct or indirect co-authors of the other author $a_2$).

### Question 1.1.7 (5 Points)

Write an SQL query that returns pairs of authors that are from the same university and have published in the same set of fields.

### Question 1.1.8 (2 Points)

Write an SQL query that returns the accumulative count of published articles for each journal over its issues.

### Question 1.1.9 (4 Points)

Write an SQL query that returns names of journal which have seen a significant drop (50%) in the number of published articles in the last three issues. That is, for each issue you have to count the number of published articles in that issue and the two preceding issues. The drop is then determined by comparing the numbers of two adjacent issues $x$ and $x + 1$.

### Question 1.1.10 Optional Bonus Question (10 Bonus Points)

Consider a relation storing information about items in a warehouse. The relation stores for each item its type and the time interval during which the item was stored in the warehouse. An example instance is shown below. Write a single SQL query that returns for each time interval how many items are in the warehouse. Note that there is some freedom in how the result is encoded. For instance, two possible equivalent query results are shown below. **Note: The result returned by the query should not contain any overlapping time intervals!**

## warehouse

| item | from | to |
|---|---|---|
| Sausage | 2020-01-01 | 2020-01-03 |
| Eggs | 2020-01-01 | 2020-01-02 |
| Bacon | 2020-01-03 | 2020-01-03 |

## query result (one option)

| cnt | from | to |
|---|---|---|
| 2 | 2020-01-01 | 2020-01-03 |

## query result (another option)

| cnt | from | to |
|---|---|---|
| 2 | 2020-01-01 | 2020-01-02 |
| 2 | 2020-01-03 | 2020-01-03 |

## Part 1.2 Relational Algebra (Total: 29 Points)

### Question 1.2.1 Relational Algebra (4 Points)

Write a relational (bag semantics) algebra expression over the schema from the SQL part (part 1) that returns universities for which at least 3 authors have published in the Science journal.

### Question 1.2.2 Relational Algebra (3 Points)

Write a relational (bag semantics) algebra expression over the schema from the SQL part (part 1) that returns the number of authors for each paper.

### Question 1.2.3 Relational Algebra (5 Points)

Write a relational (bag semantics) algebra expression over the schema from the SQL part (part 1) that returns authors that have published in all journals.

### Question 1.2.4 SQL → Relational Algebra (5 Points)

Translate the SQL query from Question 1.1.1 into relational algebra (bag semantics).

### Question 1.2.5 SQL → Relational Algebra (6 Points)

Translate the SQL query from question 1.1.2 into relational algebra (bag semantics).

### Question 1.2.6 SQL → Relational Algebra (6 Points)

Translate the SQL query from question 1.1.4 into relational algebra (bag semantics).

### Question 1.2.7 Equivalences (4 Points)

Consider the following relation schemas (primary key attributes are underlined):

$R(\underline{A}, B)$, $S(\underline{B}, C)$, $T(\underline{C}, D)$, $U(\underline{E}, F, G)$.

Furthermore, assume that $S.B$ is a foreign key to $R$ and $T.C$ is a foreign key to $S$. Check equivalences that are correct under **bag semantics**. For example $R \bowtie R \equiv R$ should be checked, whereas $R \equiv S$ should not be checked.

- ❑ $\delta(\delta(R)) \equiv \delta(R)$
- ❑ ${}_F\alpha_{sum(E)}(U) \equiv {}_F\alpha_{sum(X)}({}_{F,G}\alpha_{sum(E)\to X}(U))$
- ❑ $\delta(\pi_A(R \bowtie_{B=C} S)) \equiv \pi_A(R \bowtie_{B=C} S)$
- ❑ $R \triangleright S \equiv R - (\pi_{A,B}(R \bowtie S))$
- ❑ $\pi_B(R \bowtie S) \equiv \pi_B(R)$
- ❑ $\pi_C(S \bowtie R) \equiv \pi_C(S)$
- ❑ $\delta(\pi_A(R)) \equiv \pi_A(R)$
- ❑ $\pi_A(R \bowtie S) \equiv \pi_A(R)$

## Part 1.3 Index Structures (Total: 30 Points)

Assume that you have the following table:

**Item**

| SSN | name | age |
|---|---|---|
| 11 | Pete | 13 |
| 24 | Bob | 16 |
| 25 | John | 55 |
| 26 | Joe | 44 |
| 28 | Alice | 33 |
| 32 | Lily | 65 |
| 34 | Gertrud | 71 |
| 39 | Heinz | 12 |

### Question 1.3.1 Construction (12 Points)

Create a B+-tree for table *Item* on key *age* with $n = 2$ (up to two keys per node). You should start with an empty B+-tree and insert the keys in the order shown in the table above. Write down the resulting B+-tree after each step.

When splitting or merging nodes follow these conventions:

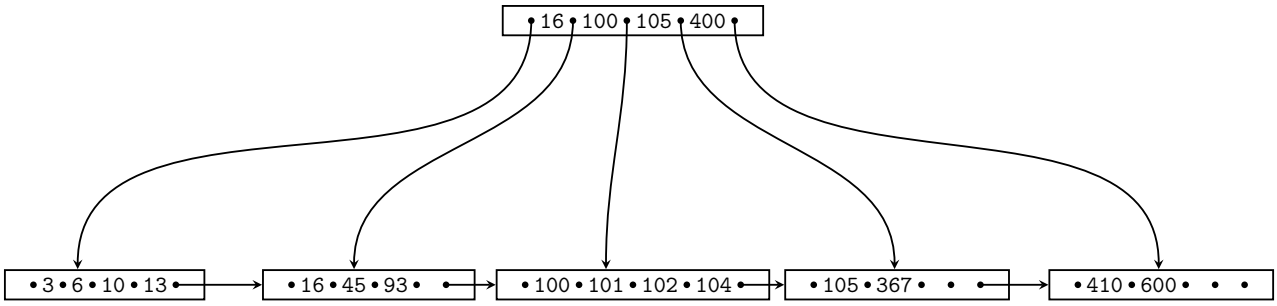
- **Leaf Split**: In case a leaf node needs to be split during insertion and $n$ is even, the left node should get the extra key. E.g, if $n = 2$ and we insert a key 4 into a node [1,5], then the resulting nodes should be [1,4] and [5]. For odd values of $n$ we can always evenly split the keys between the two nodes. In both cases the value inserted into the parent is the smallest value of the right node.
- **Non-Leaf Split**: In case a non-leaf node needs to be split and $n$ is odd, we cannot split the node evenly (one of the new nodes will have one more key). In this case the "middle" value inserted into the parent should be taken from the right node. E.g., if $n = 3$ and we have to split a non-leaf node [1,3,4,5], the resulting nodes would be [1,3] and [5]. The value inserted into the parent would be 4.
- **Node Underflow**: In case of a node underflow you should first try to redistribute values from a sibling and only if this fails merge the node with one of its siblings. Both approaches should prefer the left sibling. E.g., if we can borrow values from both the left and right sibling, you should borrow from the left one.

## Question 1.3.2 Operations (10 Points)

Given is the B+-tree shown below ($n = 4$). Execute the following operations and write down the resulting B+-tree after each operation:

**insert(95), insert(96), insert(103), delete(3), delete(6), delete(13)**

Use the conventions for splitting and merging introduced in the previous question.



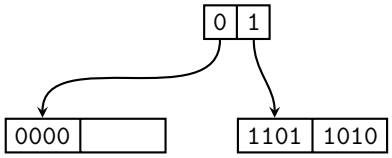
### Question 1.3.3 Extensible Hashing (8 Points)

Consider the extensible Hash index shown below that is the result of inserting values 0, 1, and 2. Each page holds two keys. Execute the following operations

`insert(4),insert(1),insert(7),insert(7)`

and write down the resulting index after each operation. Assume the hash function is defined as:

| x | h(x) |
|---|---|
| 0 | 1101 |
| 1 | 0000 |
| 2 | 1010 |
| 3 | 1100 |
| 4 | 0001 |
| 5 | 0000 |
| 6 | 1010 |
| 7 | 0111 |
| 8 | 1110 |



## Part 1.4 Result Size Estimations (Total: 10 Points)

Consider a table `lecture` with attributes `title`, `campus`, `topic`, `roomSize`, a table `student` with `name`, `major`, `age`, and a table `attendsLecture` with attributes `student`, `lecture`, and `hoursAttended`. `attendsLecture.student` is a foreign key to `student`. Attribute `lecture` of relation `attendsService` is a foreign key to of relation `lecture`. Given are the following statistics:

$$T(lecture) = 10 \qquad T(student) = 8,000 \qquad T(attendsLecture) = 40,000$$
$$V(lecture, title) = 10 \qquad V(student, name) = 7,500 \qquad V(attendsLecture, student) = 8,000$$
$$V(lecture, campus) = 2 \qquad V(student, major) = 4 \qquad V(attendsLecture, lecture) = 8$$
$$V(lecture, topic) = 3 \qquad V(student, age) = 40 \qquad V(attendsLecture, hoursAttended) = 100$$
$$V(lecture, roomSize) = 5$$

### Question 1.4.1 Estimate Result Size (3 Points)

Estimate the number of result tuples for the query $q = \sigma_{topic=DB}(lecture)$ using the first assumption presented in class (values used in queries are uniformly distributed within the active domain).

### Question 1.4.2 Estimate Result Size (3 Points)

Estimate the number of result tuples for the query $q = \sigma_{hoursAttended<20}(attendsLecture)$ using the first assumption presented in class. The minimum and maximum values of attribute *hoursAttended* are 0 and 100.

### Question 1.4.3 Estimate Result Size (4 Points)

Estimate the number of result tuples for the query $q$ below using the first assumption presented in class.

$$q = (student \bowtie_{name=student} \sigma_{hoursAttended=15}(attendsLecture) \bowtie_{lecture=title} lecture$$