Introduction to Database Systems
Trial Exam Spring 2021

(Adapted from Retake Exam August 2019)

Björn Thór Jónsson

May 14, 2021

Instructions

You have 4 hours to answer 6 problems described in the following. There are 7 problems in the exam, but problem 2 is only for BSc students and problem 3 is only for MSc students. The exam consists of 10 numbered pages. Unless instructed otherwise your answers must be provided in the LearnIT quiz *Trial Exam*.

Database description

In this exam you will work with a fictional garment database. To start working with the database, import/run `trialexam.sql` found in LearnIT using the PostgreSQL DBMS on your laptop. As the database contains nearly 200K rows, it is recommended to use `psql` for this purpose.

The database contains information on designer garments, in the following schema:

```
gDesigners(d_ID, d_Name, d_Country)
gGarments(g_ID, g_Price, d_ID, co_ID)

gTypes(t_ID, t_Name, t_Category)
gHasType(g_ID, t_ID, ht_Importance)

gFabrics(f_ID, f_Name)
gElements(f_ID, e_Element)
gMadeOf(g_ID, f_ID, mo_Percentage)
```

Most attributes are self-explanatory, with attribute names starting with the capital letter(s) of the first table they appear in. In the table `gGarments`, the nullable attribute `co_ID` references the ID of a collaborating designer. We refer to attribute `d_ID` as *main designer* and `co_ID` as *co-designer*. A designer cannot collaborate with itself.

Primary and foreign keys are correctly defined. The data, however, is sometimes incomplete. For example, the sum of `mo_Percentage` is not 100 for all garments. Furthermore, it should be noted that the data is completely fictional: while some tables are (partially) based on actual online lists, others are randomly generated.

1 SQL (40 points)

Answer each of the following questions using a single SQL query on the `garments` database:

(a) In the database, there are 25,317 garments with price higher than 20,000. How many garments are missing a price value?

(b) In the database, 257 different *main designers* have produced a garment with a fabric that a) has percentage higher than 25%, and b) contains the element 'Databasium'. How many different main designers have produced a garment with a fabric that a) has percentage higher than 25%, and b) contains the element 'Procrastinium'?

(c) How many designers have only worked alone? Meaning that they never collaborated on a garment, neither as main designer or co-designer. Hint: The answer is not 0.

(d) The average price of all garments by main designer with `d_ID` of 100 is 814057.8333. What is the `d_ID` of the main designer with the highest average price? Note that the return value of this query is a `d_ID`, not a count; in a different database instance your query might thus return more than one row, but in this instance it should only return one row.

(e) How many different elements, with a name starting with 'C', appear in 5 or more fabrics?

(f) For a garment to have correct data about its composition, there should be some entries in `gMadeOf` for that garment with `mo_Percentage` values that add up to 100. For how many garments in the database is data about their composition not correct?

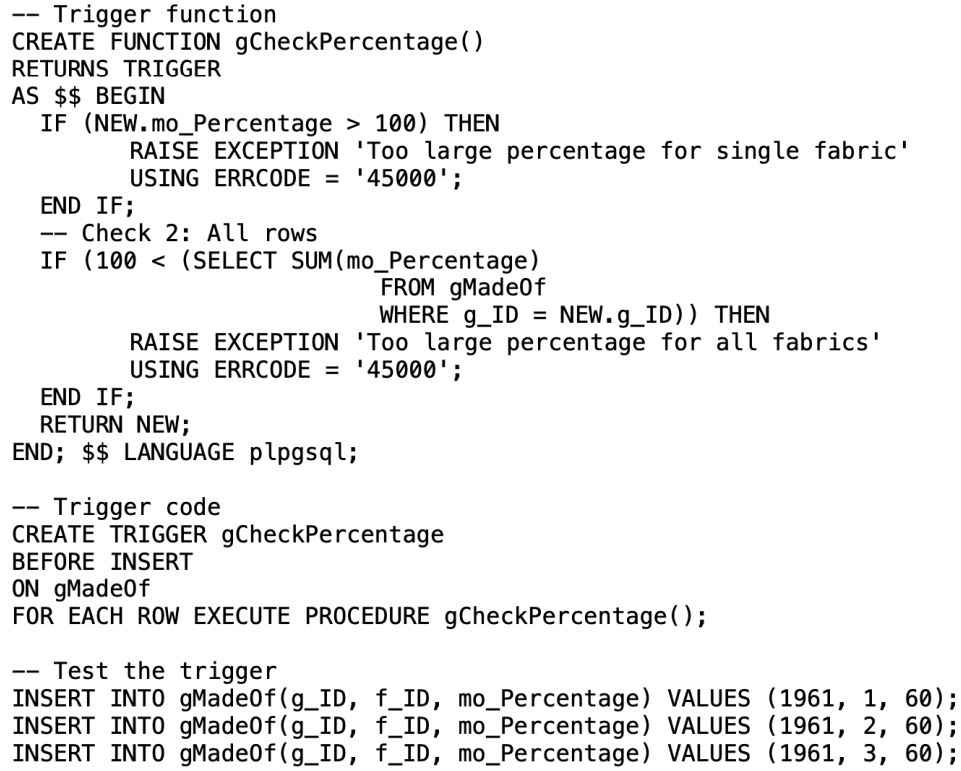
(g) In the database, there are 16 different garment types of category 'Upper'. Not surprisingly, no main designer has designed a garment in all these categories. There are, however, some main designers that have designed at least one garment of all types in the category 'Dress'. How many such main designers are in the database? Note that in this *division* query, you should consider only the main designers, not co-designers.

(h) In the database, designer with `d_ID` of 100 has collaborated, either as main designer or as co-designer, with 13 other designers from 7 different countries. Which designer has collaborated with other designers from the highest number of different countries? Note that the return value of this query is a `d_ID`, not a count.

Enter each query, along with its numerical answer, in LearnIT. Queries must work for any database instance. Queries should not return anything except the answer; a query that returns more information will not receive full points, even if the answer is part of the returned result. A sequence of several queries that answer the question will not receive full points, but subqueries and views can be used. Queries should be as simple as possible; queries that are unnecessarily complex may not get full marks, despite returning the correct answer. If you are unable to complete the query you can still submit your attempt, along with a brief description of the problem with the query, and it may be given partial points.

2 (BSc ONLY) SQL programming (5 points)

To answer this question, you will need to study the `gMadeOf` relation of the `garments` database. Consider the SQL trigger code in Figure 1 (it is an image, so the code cannot be copied from the PDF).



```
-- Trigger function
CREATE FUNCTION gCheckPercentage()
RETURNS TRIGGER
AS $$ BEGIN
  IF (NEW.mo_Percentage > 100) THEN
        RAISE EXCEPTION 'Too large percentage for single fabric'
        USING ERRCODE = '45000';
  END IF;
  -- Check 2: All rows
  IF (100 < (SELECT SUM(mo_Percentage)
                          FROM gMadeOf
                          WHERE g_ID = NEW.g_ID)) THEN
        RAISE EXCEPTION 'Too large percentage for all fabrics'
        USING ERRCODE = '45000';
  END IF;
  RETURN NEW;
END; $$ LANGUAGE plpgsql;

-- Trigger code
CREATE TRIGGER gCheckPercentage
BEFORE INSERT
ON gMadeOf
FOR EACH ROW EXECUTE PROCEDURE gCheckPercentage();

-- Test the trigger
INSERT INTO gMadeOf(g_ID, f_ID, mo_Percentage) VALUES (1961, 1, 60);
INSERT INTO gMadeOf(g_ID, f_ID, mo_Percentage) VALUES (1961, 2, 60);
INSERT INTO gMadeOf(g_ID, f_ID, mo_Percentage) VALUES (1961, 3, 60);
```

Figure 1: Insertion trigger `gCheckPercentage` for the `gMadeOf` relation.

a) Select the true statements:

(a) Check 1 can be replaced by a CHECK constraint on the `garments` relation.

(b) Check 2 can be replaced by a CHECK constraint on the `garments` relation.

b) If the `gMadeOf` relation is empty when the three INSERT statements are issued, while the relevant garment and fabrics exist, which INSERT statement will be the first to give an error:

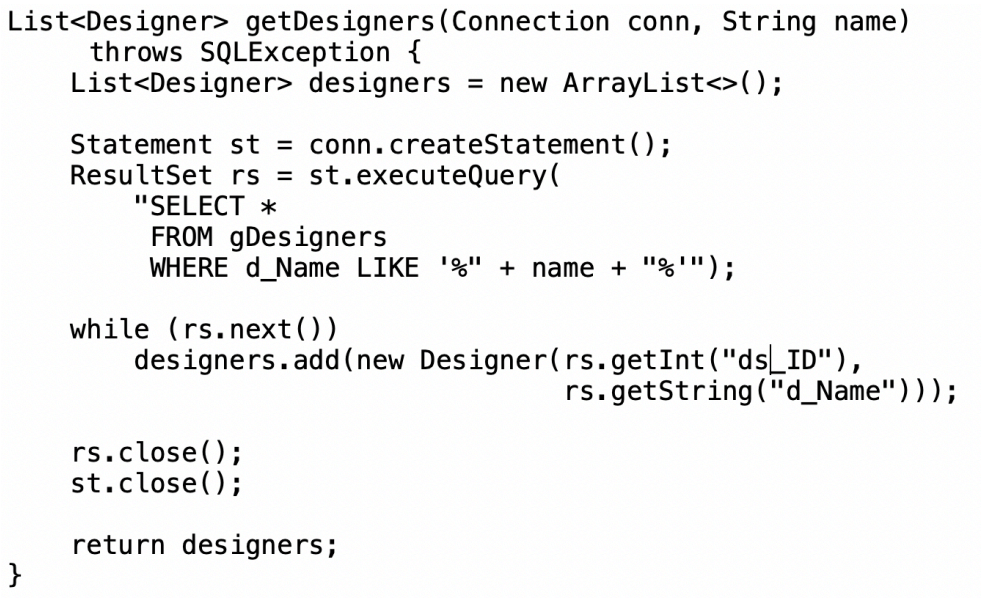
(a) The first INSERT statement will be the first to give an error.

(b) The second INSERT statement will be the first to give an error.

(c) The third INSERT statement will be the first to give an error.

3 (MSc ONLY) Database programming (5 points)

To answer this question, you will need to study the `gDesigners` relation of the `garments` database. Consider the Java code in Figure 2 (it is an image, so the code cannot be copied from the PDF).



```
List<Designer> getDesigners(Connection conn, String name)
     throws SQLException {
    List<Designer> designers = new ArrayList<>();

    Statement st = conn.createStatement();
    ResultSet rs = st.executeQuery(
        "SELECT *
         FROM gDesigners
         WHERE d_Name LIKE '%" + name + "%'");

    while (rs.next())
        designers.add(new Designer(rs.getInt("ds_ID"),
                                   rs.getString("d_Name")));

    rs.close();
    st.close();

    return designers;
}
```

Figure 2: Query for designers from the `gDesigners` relation.

a) Select the true statements:

(a) The code is safe against SQL injection attacks.

(b) The code terminates the connection to the database after the statement is closed.

(c) Using executeQuery will throw an exception for a SELECT query.

(d) The code is using Object Relational Mapping.

b) If the `gDesigners` relation is empty when the function is called, which of the following will happen:

(a) The code will not compile.

(b) The statement will throw an exception.

(c) An empty list will be returned.

4 ER Diagrams and Normalization (25 points)

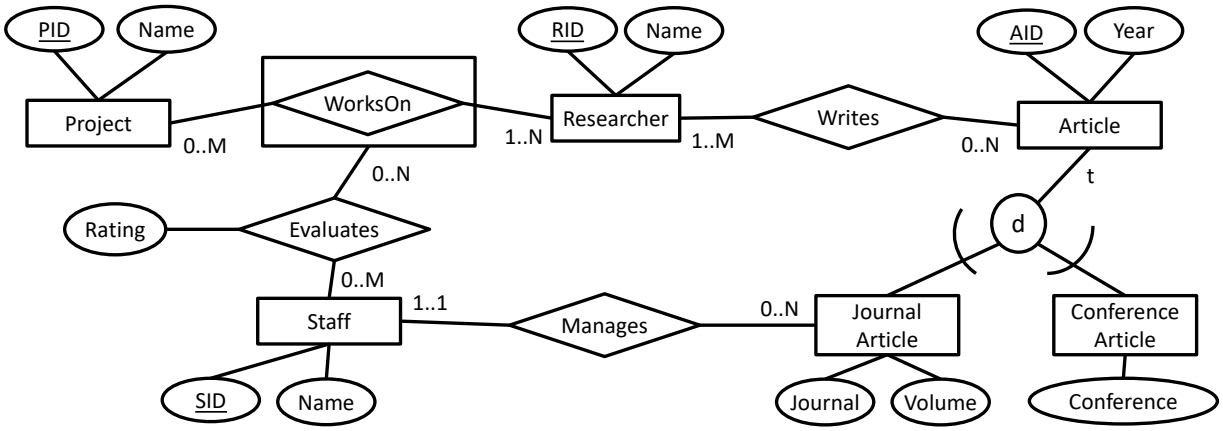


Figure 3: ER Diagram for research database.

a) The ER diagram in Figure 3 shows a simple research database. Select the true statements. You should base your answers **only** on the ER diagram:

(a) A researcher must write at least one article.

(b) An article can be both a journal and a conference article.

(c) The same real-life person can be both a researcher or a staff member.

(d) When multiple researchers work on a project, one must be clearly marked as the project leader.

(e) The resulting database should have exactly 8 tables.

(f) The table for the Evaluates relationship should have foreign key relationships with 2 tables.

b) Write SQL DDL commands to create the research database based on the ER diagram in Figure 3. The DDL script *must run* in PostgreSQL as a whole. The relations must include all primary key and foreign key constraints. Constraints that cannot be enforced with standard primary key and foreign key constraints can be omitted. Make reasonable assumptions on the attribute types.

c) Write an ER diagram for a medical database. The diagram should clearly show the entities, relationships and participation constrains described below. Use the notation presented in the textbook and lectures. Attributes are not important. If you need to make additional assumptions put them in the box below.

- All medical staff members are either doctors or nurses, not both.
- Each patient is treated by one doctor.
- Each patient can be cared for by multiple nurses.
- Each patient must be insured by at least one insurance company.
- Insurance companies may review the treatment of patients.

d) Consider a table $R(A, B, C, D, E)$ with the following dependencies:

$$\begin{aligned} AB &\rightarrow CDE \\ A &\rightarrow C \\ D &\rightarrow B \\ A &\rightarrow B \end{aligned}$$

Select the true statements:

(a) AB is the only (candidate) key of R.

(b) $A \rightarrow C$ is a trivial functional dependency.

(c) Normalizing to 3NF or BCNF results in exactly two relations.

(d) The relation can be normalized to BCNF without losing dependencies.

e) Consider a table $R(A, B, C, D, E)$ with the following dependencies:

$$\begin{aligned} AB &\rightarrow CD \\ A &\rightarrow E \\ C &\rightarrow A \\ ABC &\rightarrow ADE \end{aligned}$$

Normalize R to the highest possible normal form (3NF or BCNF), based on functional dependencies, while allowing all functional dependencies (excluding trivial, unavoidable, and derivable dependencies) to be checked within a single relation. For each resulting relation, write its columns and clearly indicate whether it is in BCNF.

5 Index Selection (10 points)

Consider the following relation with information on passenger airplanes:

Airplanes(id, name, year, passengers, <many long attributes>)

The attributes id, year and passengers are integer values, while name is a string, and none of these are nullable. Assume that year is uniformly distributed over the last 100 years, and passengers is always greater than 0. Now consider the following three SQL queries:

Query 1

```
select id, name
from Airplanes
where name like '%737%';
```

Query 2

```
select *
from Airplanes
where passengers > 0
order by passengers;
```

Query 3

```
select id
from Airplanes
where year = 1992;
```

Answer each of the following questions:

(a) Indicate for each query whether a clustered index should be defined (i.e., would be preferable to a non-clustered index or no index at all). Explain your answer and define the indexes you consider.

(b) Indicate for each query whether a covering index could be defined (i.e., would be preferable to a clustered index). Explain your answer and define the indexes you consider.

(c) Considering all three queries, which clustered index would you define on Airplanes? Explain your answer.

6 DBMS Architecture (10 points)

a) Argue why Write-Ahead Logging is appropriate for HDD-based DBMS.

b) How is the cost of queries evaluated in HDD-based DBMS? Select all statements that apply:

(a) It is only the cost of I/O.

(b) It is only CPU cost.

(c) It is a mix of CPU and I/O cost.

(d) It is fixed.

7 Transactions (10 points)

a) Define transactions.

b) Why is steal/no-force buffer management favoured in commercial DBMS? Select all statements that apply:

(a) Because it is the simplest solution.

(b) Because sequential writes are much faster than random writes. (T)

(c) Because it avoids situations where the DBMS would run out of memory. (T)

(d) Because it minimizes software overhead.

Trial Exam Fall 2020

1. **1a) SQL**

   1a) Write your SQL query here:

   Notes: (not included in XML)

   - `See Q1-SQL.sql`

2. **1a) Numerical answer**

   1a) Run the query of the previous question and paste the result here (an integer):

   - 2853 ✓

3. **1b) SQL**

   1b) Write your SQL query here:

   Notes: (not included in XML)

   - `See Q1-SQL.sql`

4. **1b) Numerical answer**

   1b) Run the query of the previous question and paste the result here (an integer):

   - 250 ✓

5. **1c) SQL**

   1c) Write your SQL query here:

   Notes: (not included in XML)

   - `See Q1-SQL.sql`

6. **1c) Numerical answer**

   1c) Run the query of the previous question and paste the result here (an integer):

   - 100 ✓
   - 94 ✓

7. **1d) SQL**

   1d) Write your SQL query here:

   Notes: (not included in XML)

   - `See Q1-SQL.sql`

8. **1d) Numerical answer**

   1d) Run the query of the previous question and paste the result here (an identifier):

- 1481 ✓

9. **1e) SQL**

   1e) Write your SQL query here:

   Notes: (not included in XML)

   - `See Q1-SQL.sql`

10. **1e) Numerical answer**

    1e) Run the query of the previous question and paste the result here (an integer):

    - 2 ✓

11. **1f) SQL**

    1f) Write your SQL query here:

    Notes: (not included in XML)

    - `See Q1-SQL.sql`

12. **1f) Numerical answer**

    1f) Run the query of the previous question and paste the result here (an integer):

    - 5777 ✓

13. **1g) SQL**

    1g) Write your SQL query here:

    Notes: (not included in XML)

    - `See Q1-SQL.sql`

14. **1g) Numerical answer**

    1g) Run the query of the previous question and paste the result here (an integer):

    - 6 ✓

15. **1h) SQL**

    1h) Write your SQL query here:

    Notes: (not included in XML)

    - `See Q1-SQL.sql`

16. **1h) Numerical answer**

    1h) Run the query of the previous question and paste the result here (an identifier):

- 582 ✓

17. **2a) [BSc only] SQL programming**

    2a) [BSc only] Select the true statements:

    (a) Check 1 can be replaced by a CHECK constraint on the `garments` relation. (100%)
    (b) Check 2 can be replaced by a CHECK constraint on the `garments` relation. (0%)

18. **2b) [BSc only] SQL programming**

    2b) [BSc only] Select the correct choice:

    (a) The first INSERT statement will give an error.
    (b) The second INSERT statement will give an error.
    (c) The third INSERT statement will give an error. ✓

19. **3a) [MSc only] Java programming**

    3a) [MSc only] Select the true statements:

    (a) The code is safe against SQL injection attacks. (0%)
    (b) The code terminates the connection to the database after the statement is closed. (0%)
    (c) Using executeQuery will throw an exception for a SELECT query. (0%)
    (d) The code is using Object Relational Mapping. (100%)

20. **3b) [MSc only] Java programming**

    3b) [MSc only] Select the correct choice:

    (a) The code will not compile.
    (b) The statement will throw an exception.
    (c) An empty list will be returned. ✓

21. **4a) Research ER diagram**

    4a) Select the true statements:

    (a) A researcher must write at least one article. (0%)
    (b) An article can be both a journal and a conference article. (0%)
    (c) The same real-life person can be both a researcher or a staff member. (50%)
    (d) When multiple researchers work on a project, one must be clearly marked as the project leader. (0%)
    (e) The resulting database should have exactly 8 tables. (0%)
    (f) The table for the Evaluates relationship should have foreign key relationships with 2 tables. (50%)

22. **4b) DDL**

    4b) Write your DDL for creating the database. You can also write any extra assumptions, attributes or explanations you feel are necessary.

    Notes: (not included in XML)

    - `See Q4b-DDL.sql`

23. **4c) ER-diagram creation**

    4c) Upload the ER diagram.

    Notes: (not included in XML)

    - `See Q4c-ER.jpg`

24. **4d) Normalisation**

    4d) Select the true statements:

    (a) AB is the only key of R. (0%)
    (b) $A \rightarrow C$ is a trivial functional dependency. (0%)
    (c) Normalizing to 3NF or BCNF results in exactly two relations. (50%)
    (d) The relation can be normalized to BCNF without losing dependencies. (50%)

25. **4e) Normalisation**

    4e) Write down the normalized relations. For each resulting relation, write its columns and clearly indicate whether it is in BCNF.

    Notes: (not included in XML)

    - `R1(A, E) in BCNF, R2(A, B, C, D) not in BCNF.`
      `A bit of explanation: ABC --> ADE is both redundant and unavoidable (AB is key), and thus can be omitted. A --> E breaks 2NF and up and thus requires decomposition. C --> A only breaks BCNF and does not require decomposition. After decomposition, R1 is in BCNF, with key A, while R2 is in 3NF with keys AB and BC.`

26. **5a) Clustering Indexes**

    5a) Argue for a clustered index, compared to unclustered or no index.

    Notes: (not included in XML)

    - `(Q1) Since the LIKE expression starts with a %, a clustered index cannot be used to reduce the number of rows that must be looked at, and therefore gives no benefit over reading the whole relation. (Q2) Since the query has ordering and a where clause on the same attribute, a clustered index on that Airplanes(passengers) would allow to read the relation sequentially and return the results in the correct order. With an`

unclustered index, a random disk IOs would to needed to read each record, which would be much more expensive. If there is no index, the relation would need to be sorted, in O(n log n) time, which is also more expensive. (Q3) Since each query would return on average 1% of the relation, an index on Airplanes(year) would be better than no index, and a clustered index would perform better than an unclustered index, due to its sequential reads.

27. **5b) Covering Indexes**

    5b) Argue for a covering index for each query, compared to clustered index.

    Notes: (not included in XML)

    - (Q1) A covering index on Airplanes(name, id) could be used to reduce the number of columns read, and hence reduce the overall work to process the query. (Q2) Since all attributes are returned by this query, a covering index would need to replicate the whole relation, and hence a clustered index is a better choice. (Q3) A covering index on Airplanes(year, id) would be optimal for this query, as no random IOs would be needed to read the actual records.

28. **5c) Best Clustered Index**

    Considering all three queries, explain which clustered index would you define.

    Notes: (not included in XML)

    - Since (a) covering indexes can be used effectively for queries Q1 and Q3, but not for Q2, while (b) a clustered index is very effective for Q2, the best clustered index for these three queries would be on Airplanes(pass

29. **6a) DBMS Architecture**

    6a) Write your reflections here:

    Notes: (not included in XML)

    - The crux of the answer should be: WAL writes sequentially to a log, which is much faster than randomly writing data in-place.

30. **6b) DBMS Architecture**

    6b) Select the true statements:

    (a) It is only the cost of I/O. (100%)
    (b) It is only CPU cost. (0%)
    (c) It is a mix of CPU and I/O cost. (0%)
    (d) It is fixed. (0%)

31. **7a) Transactions**

    7a) Write your definition here:

    Notes: (not included in XML)

    - `The crux of your answer should be: A group of operations for which the DBMS provides some guarantees.`

32. **7b) Transactions**

    7b) Select the true statements:

    (a) Because it is the simplest solution. (0%)
    (b) Because sequential writes are much faster than random writes. (50%)
    (c) Because it avoids situations where the DBMS would run out of memory. (50%)
    (d) Because it minimizes software overhead. (0%)