Introduction to Database Systems
MSc and BSc Final Exams

Björn Thór Jónsson

December 17, 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 11 numbered pages. Unless instructed otherwise your answers must be provided in the LearnIT quiz *Final Exam December 2021*.

Database Description for Questions 1–3

In this exam you will work with a fictional (and poorly designed!) database of recipes. To start working with the database, run the commands in `idb-december-2021.sql` found in LearnIT using the PostgreSQL DBMS on your laptop. It is recommended to use `psql` for this purpose. The database has the following schema:

```
cuisines (id, name)
ingredients (id, name, type)
belong_to (ingredient_id, cuisine_id)

chefs (id, name)
recipes (id, name, belong_to, created_by)
master (chef_id, recipe_id)

steps (recipe_id, step, description)
use (recipe_id, step, ingredient_id, quantity, unit)
```

Primary keys and foreign keys are defined and attributes are largely self-explanatory. You may study the DDL commands to understand the details of the tables (the CREATE TABLE statements are at the top of the script), consider the ER-diagram in Figure 1, or inspect the tables using SQL queries. Following are some additional notes that are important for your queries:

- There are two `belong_to` relationships, one between recipes and cuisines, and the other between ingredients and cuisines. Recipes belong to at most one cuisine, while ingredients may belong to many.
- One chef has created each recipe, while many may master the same recipe.
- While the database has quantity in different units in the `use` table, we ignore those units in this exam.

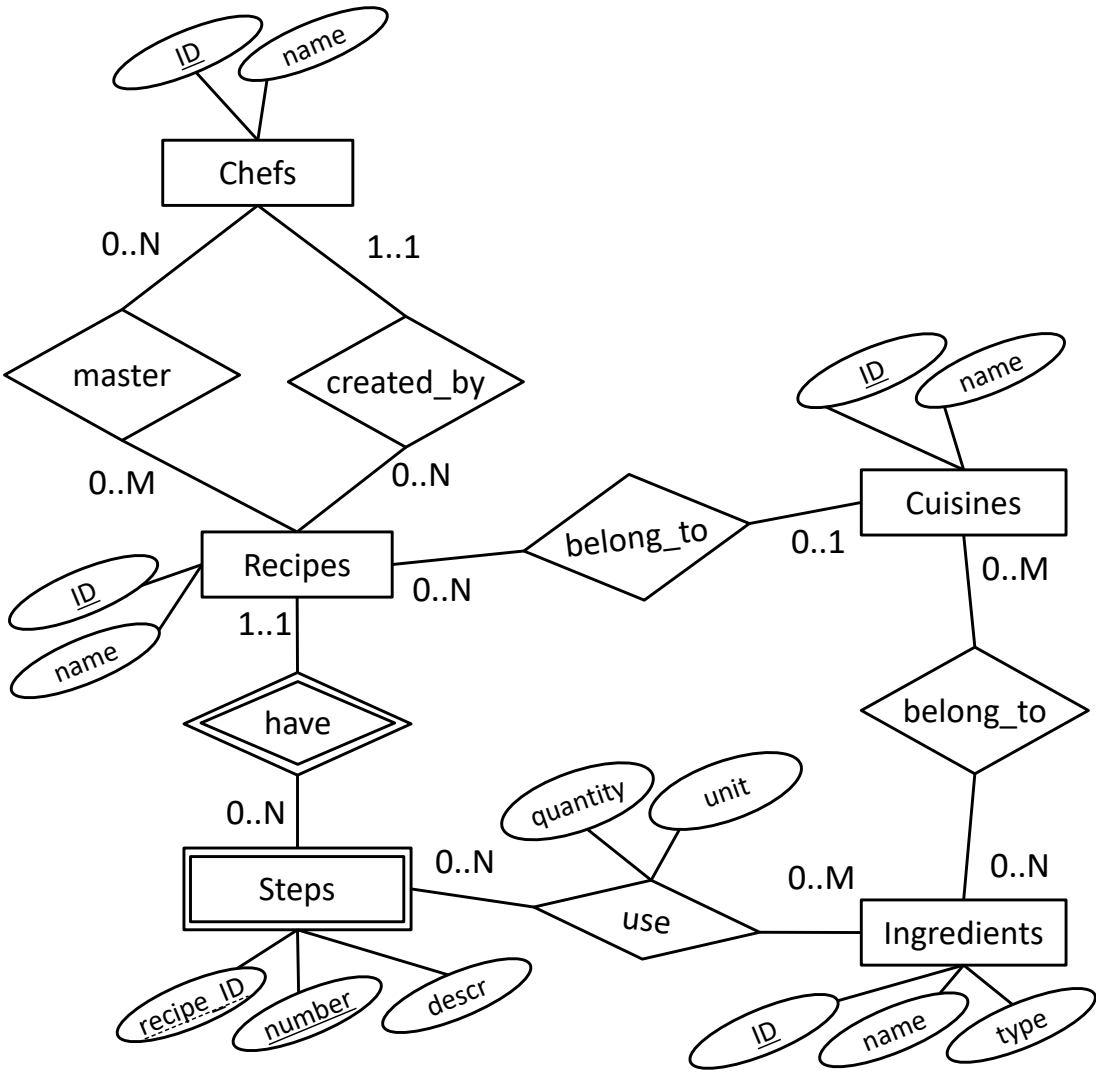


Figure 1: ER Diagram for the recipe database.

Instructions for SQL Queries in Question 1

Queries must return correct results for any database instance. They should avoid system-specific features, including the LIMIT keyword. 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, and it may be given partial points.

1 SQL (40 points)

Answer each of the following questions using a single SQL query on the examination database. Enter each query, along with its numerical answer, in LearnIT. Queries must adhere to the detailed guidelines given on Page 3.

(a) A total of 410 chefs have created at least one recipe. How many have not created any recipes?

(b) The chef 'Foodalicious' has mastered 56 recipes that have some ingredient(s) of type 'spice'. How many recipes that have some ingredient(s) of type 'spice' has the chef 'Spicemaster' mastered?

(c) There are 1,257 recipes in the database with 10 or more steps registered. How many recipes have 3 or fewer steps registered?

(d) How many recipes belong to the same cuisine as at least one of their ingredients?

(e) The recipe with name 'Fresh Tomato Salsa Restaurant-Style' has the most steps of all recipes, or 38. What is/are the name of the recipe/s with the most different ingredients of all recipes?

Note: The output of this query is a set of one or more recipe names.

(f) We define the *spice ratio* of a cuisine as the number of ingredients that belong to it that are of type 'spice' divided by the total number of ingredients that belong to the cuisine. Here we consider only cuisines that actually have spices. The highest spice ratio is 1.0, and this spice ratio is shared by 8 cuisines. How many cuisines share the lowest spice ratio?

(g) There are 4,169 recipes that contain some ingredient of all ingredient types. How many recipes contain some ingredient of all ingredient types in the same step?

Note: This is a division query; points will only be awarded if division is attempted.

(h) Write a query that outputs the `id` and `name` of chefs, and total ingredient quantity (regardless of units), in order of decreasing quantity, for chefs that have created recipes in a cuisine with 'Indian' in the name, but only considering ingredients that belong to a cuisine with 'Thai' in the name.

2 (BSc ONLY) SQL programming (5 points)

Consider the SQL trigger code in Figure 2 (it is an image, so the code cannot be copied from the PDF). The goal of the trigger is to ensure quality of ingredients listing in recipes. To answer this question, you will need to study the database instance.

```
CREATE FUNCTION CheckCuisines() RETURNS TRIGGER
AS $$ BEGIN
  -- Check 1: Is the quantity OK?
  IF (NEW.quantity <= 0) THEN
    NEW.quantity = 1;
  END IF;
  -- Check 2: Is the ingredient in the correct cuisine?
  IF NOT EXISTS (
      SELECT *
      FROM recipes R
        JOIN belong_to B on R.belong_to = B.cuisine_id
      WHERE R.id = NEW.recipe_id
        AND B.ingredient_ID = NEW.ingredient_id) THEN
    RAISE EXCEPTION 'There is no common cuisine'
    USING ERRCODE = '45000';
  END IF;
  RETURN NEW;
END; $$ LANGUAGE plpgsql;

CREATE TRIGGER CheckCuisines
AFTER INSERT ON use
FOR EACH ROW EXECUTE PROCEDURE CheckCuisines();

INSERT INTO use(recipe_id, step, ingredient_id, quantity, unit) VALUES (1, 1, 33, -1, 'lb');
INSERT INTO use(recipe_id, step, ingredient_id, quantity, unit) VALUES (1, 1, 34, -1, 'lb');
INSERT INTO use(recipe_id, step, ingredient_id, quantity, unit) VALUES (1, 1, 34,  1, 'lb');
```

Figure 2: Insertion trigger `CheckCuisines` for the `use` relation.

Select the true statements:

(a) The trigger prevents negative quantity values from entering the relation.

(b) The trigger ensures that only ingredients that belong to the same cuisine as the recipe can be used.

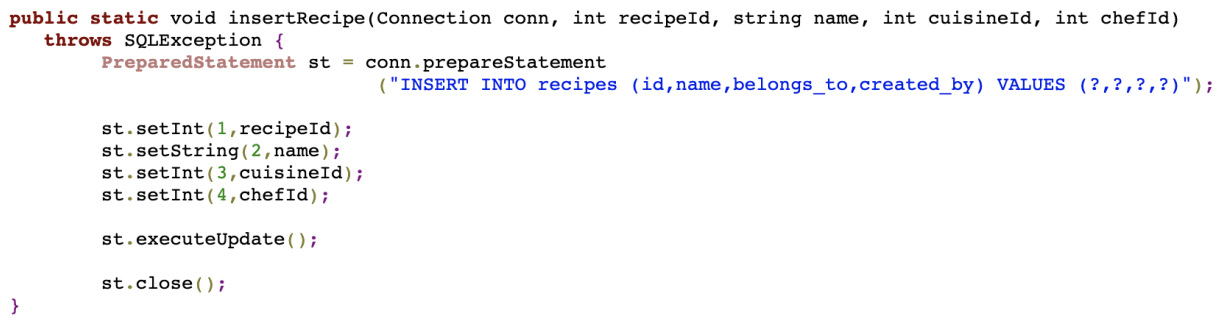
(c) The first INSERT will succeed in adding a row to the table.

(d) The second INSERT will succeed in adding a row to the table.

(e) The third INSERT will succeed in adding a row to the table.

3 (MSc ONLY) Database programming (5 points)

To answer this question, you will need to study the `recipes` relation of the food database. Consider the Java code in Figure 3 (it is an image, so the code cannot be copied from the PDF).



```
public static void insertRecipe(Connection conn, int recipeId, string name, int cuisineId, int chefId)
   throws SQLException {
        PreparedStatement st = conn.prepareStatement
                               ("INSERT INTO recipes (id,name,belongs_to,created_by) VALUES (?,?,?,?)");

        st.setInt(1,recipeId);
        st.setString(2,name);
        st.setInt(3,cuisineId);
        st.setInt(4,chefId);

        st.executeUpdate();

        st.close();
}
```

Figure 3: Insertion code for `recipes` relation.

Select the true statements:

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

(b) The code should be using transactions to preserve the integrity of the database.

(c) The code is not using Object Relational Mapping.

(d) The method call `insertRecipe(0,"paella",0,0)` will handle freeing the resources associated to the statement.

(e) By default the recipe will not be inserted into the database until `conn.commit()` is called.

4 ER Diagrams and Normalization (25 points)

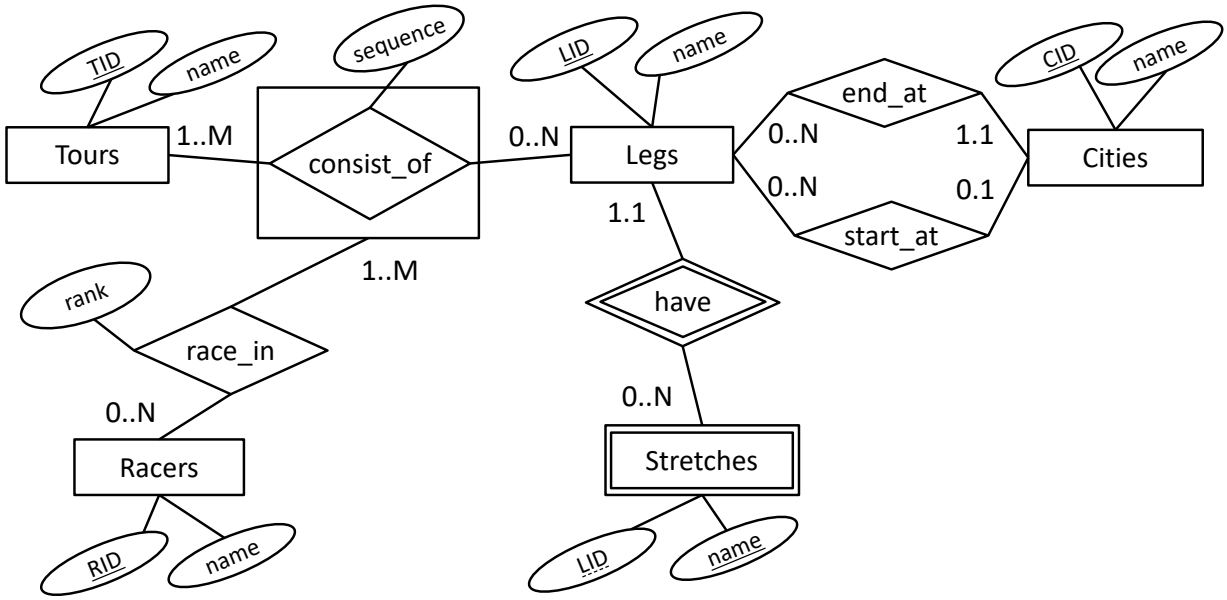


Figure 4: ER Diagram for a bicycle race database.

a) The ER diagram in Figure 4 shows a simple database for bicycle races. Select the true statements. You should base your answers **only** on the ER diagram:

(a) A racer has competed in at most 1 tour.

(b) A racer has competed in at least 1 tour.

(c) Many stretches can have the same name.

(d) Many racers can have the same name.

(e) Each stretch connects to at least one racer, via the relationships.

(f) All cities are the start of some leg.

b) Write SQL DDL commands to create a bicycle race database based on the ER diagram in Figure 4. 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. The type of name attributes should be string, all other attributes should be int.

c) Write an ER diagram for a database of television series based on the following requirements. The diagram should clearly show the entities, attributes, relationships and participation constrains described below. Use the notation presented in the textbook and lectures. Attributes are only important if mentioned. If you need to make additional assumptions put them in the box below.

- Series are either Original or Compilation, not both. Each type has multiple attributes, but you can ignore them in your design.
- Each series is produced by one Producer.
- Each episode is part of one Original series.
- Episodes can also be part of many Compilation series.
- Actors can act in multiple episodes. Each episode has at least one actor.
- Producers rate the acting of actors in individual episodes, with a rating attribute.

d) Consider a table $R(L, M, N, O, P)$ with the following dependencies:

$$\begin{aligned} LMN &\rightarrow OP \\ L &\rightarrow O \\ P &\rightarrow M \\ N &\rightarrow N \end{aligned}$$

Select the true statements:

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

(b) $LMNO \rightarrow P$ is an unavoidable 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 $R(L, M, N, O, P)$ with the following dependencies:

$$\begin{aligned} L &\rightarrow MN \\ N &\rightarrow O \\ MN &\rightarrow P \end{aligned}$$

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 large relation with information on products:

Products(id, name, descr, year, value, <many long attributes>)

The attributes id, year and value are integer values, while name and descr are strings, and none of these are nullable. Assume that name is a candidate key and value ranges uniformly from 0 to 5000. Now consider the following three SQL queries:

Query 1

```
select *
from Products
where name = 'The Best Product in the World';
```

Query 2

```
select *
from Products
where value > 2000
```

Query 3

```
select year, avg(value)
from Products
group by year;
```

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 Products? Explain your answer.

6 Hardware and DBMS Design (10 points)

a) Select the correct statements below:

(a) In a main-memory relational database system, it remains necessary to log the "old" values during updates, in case the transaction aborts.

(b) NoSQL literally means that SQL-like query languages must not be supported by NoSQL systems.

(c) Document stores cannot store business data that is typically stored in relational systems.

(d) While locking whole tables guarantees serializability in a relational system, it leads to terrible performance, which is why it is turned off by default in most systems.

b) Discuss pros and cons of storing multiple copies of data in a geographically distributed database, compared to storing a single copy. Be sure to mention at least one advantage and one disadvantage.

7 Data Systems for Analytics (10 points)

a) Select the correct statements below:

(a) All big data applications are evil in nature.

(b) Spark is designed for supporting user-facing and interactive applications.

(c) Many big data applications require so much computation that a distributed processing framework is necessary.

(d) A machine learning model that is based on real data can never have any bias, because it accurately models real life.

b) What is the the most important disk access pattern of typical big data applications? Support your answers with arguments.

Final Exam December 2021

1. **1a) SQL**

   ESSAY | 4 points | 0.10 penalty | monospaced

   1a) Write your SQL query here:

   Notes for grader:

   - `See idb-august-2021-SQL.sql`

2. **1a) Query answer**

   NUMERICAL | 1 point | 0.10 penalty

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

   - 6 ✓

3. **1b) SQL**

   ESSAY | 4 points | 0.10 penalty | monospaced

   1b) Write your SQL query here:

   Notes for grader:

   - `See idb-august-2021-SQL.sql`

4. **1b) Query answer**

   NUMERICAL | 1 point | 0.10 penalty

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

   - 57 ✓

5. **1c) SQL**

   ESSAY | 4 points | 0.10 penalty | monospaced

   1c) Write your SQL query here:

   Notes for grader:

   - `See idb-august-2021-SQL.sql`

6. **1c) Query answer**

   NUMERICAL | 1 point | 0.10 penalty

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

   - 1149 ✓

7. **1d) SQL**

   ESSAY | 4 points | 0.10 penalty | monospaced

   1d) Write your SQL query here:

   Notes for grader:

   - `See idb-august-2021-SQL.sql`

8. **1d) Query answer**

   SHORTANSWER | 1 point | 0.10 penalty | Case-Insensitive

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

   - 882 ✓

9. **1e) SQL**

   ESSAY | 4 points | 0.10 penalty | monospaced

   1e) Write your SQL query here:

   Notes for grader:

   - `See idb-august-2021-SQL.sql`

10. **1e) Query answer**

    SHORTANSWER | 1 point | 0.10 penalty | Case-Insensitive

    1e) Run the query of the previous question and paste the result here (one or more names):

    - `Dinengdeng` ✓
    - `"Dinengdeng"` ✓

11. **1f) SQL**

    ESSAY | 4 points | 0.10 penalty | monospaced

    1f) Write your SQL query here:

    Notes for grader:

    - `See idb-august-2021-SQL.sql`

12. **1f) Query answer**

    NUMERICAL | 1 point | 0.10 penalty

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

    - 7 ✓

13. **1g) SQL**

ESSAY | 4 points | 0.10 penalty | monospaced

1g) Write your SQL query here:

Notes for grader:

- `See idb-august-2021-SQL.sql`

14. **1g) Query answer**

NUMERICAL | 1 point | 0.10 penalty

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

- 2722 ✓

15. **1h) SQL**

ESSAY | 5 points | 0.10 penalty | monospaced

1h) Write your SQL query here:

Notes for grader:

- `See idb-august-2021-SQL.sql`

16. **1h) Query answer**

ESSAY | 0 points | 0.10 penalty | editor

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

Notes for grader:

- `See idb-august-2021-SQL.sql`

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

MULTI | 5 points | 0.10 penalty | Multiple

2) [BSc only] Select the true statements:

(a) The trigger prevents negative quantity values from entering the relation. (−33.33333%)
(b) The trigger ensures that only ingredients that belong to the same cuisine as the recipe can be used. (50%)
(c) The first INSERT will succeed in adding a row to the table. (50%)
(d) The second INSERT will succeed in adding a row to the table. (−33.33333%)
(e) The third INSERT will succeed in adding a row to the table. (−33.33333%)

18. **3) [MSc only] Java programming**

MULTI | 5 points | 0.10 penalty | Multiple

3) [MSc only] Select the true statements:

(a) The code is safe against SQL injection attacks. (50%)
(b) The code should be using transactions to preserve the integrity of the database. (−33.33333%)
(c) The code is not using Object Relational Mapping. (50%)
(d) The method call `insertRecipe(0,"paella",0,0)` will handle freeing the resources associated to the statement. (−33.33333%)
(e) By default the recipe will not be inserted into the database until `conn.commit()` is called. (−33.33333%)

19. **4a) ER Diagram Interpretation**

MULTI | 6 points | 0.10 penalty | Multiple

4a) Select the true statements:

(a) A racer has competed in at most 1 tour. (−33.33333%)
(b) A racer has competed in at least 1 tour. (33.33333%)
(c) Many stretches can have the same name. (33.33333%)
(d) Many racers can have the same name. (33.33333%)
(e) Each stretch connects to at least one racer, via the relationships. (−33.33333%)
(f) All cities are the start of some leg. (−33.33333%)

20. **4b) SQL DDL**

ESSAY | 6 points | 0.10 penalty | monospaced

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

Notes for grader:

- `See idb-august-2021-DDL.sql.`

21. **4c) ER Diagram Creation**

ESSAY | 7 points | 0.10 penalty | monospaced

4c) Upload the ER diagram or deliver a hand drawing at the exam.

Notes for grader:

- `See idb-august-2021-ER.jpg.`

22. **4d) Normalisation**

MULTI | 3 points | 0.10 penalty | Multiple

4d) Select the true statements:

(a) LMN is the only (candidate) key of R. (−50%)
(b) $LMNO \rightarrow P$ is an unavoidable functional dependency. (50%)
(c) Normalizing to 3NF or BCNF results in exactly two relations. (50%)
(d) The relation can be normalized to BCNF without losing dependencies. (−50%)

23. **4e) Normalisation**

ESSAY | 3 points | 0.10 penalty | monospaced

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

Notes for grader:

- `LMN in BCNF (key = L), NO in BCNF (key = N), MNP in BCNF (key = MN).`

24. **5a) Clustering Indexes**

ESSAY | 4 points | 0.10 penalty | monospaced

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

Notes for grader:

- `(Q1) Since this query retrieves exactly one record, (a) an index on name will be better than no index, but (b) an unclustered index will perform as well as a clustered index.`
- `(Q2) A clustered index on value will perform well. Since the query returns about 60% of the relation, it will perform about 40% better than a full table scan, and much better than an unclustered index.`
- `(Q3) A clustered index on year would allow to answer the query with a single sequential scan. An unclustered index would perform worse (more IOs, plus random IOs), while no index would result in sorting or hashing.`

25. **5b) Covering Indexes**

ESSAY | 4 points | 0.10 penalty | monospaced

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

Notes for grader:

- `(Q1/Q2) Since whole rows are returned, covering indexes don't apply here.`
- `(Q3) A covering index on year and value would allow reading much less data than using the clustered index, even if the number of rows is the same, and hence it is optimal.`

26. **5c) Best Clustered Index**

ESSAY | 2 points | 0.10 penalty | monospaced

5c) Considering all three queries, explain which clustered index would you define.

Notes for grader:

- `While Q2 and Q3 could both benefit from a clustered index, query Q3 would be even better with a covering index, while the clustered index is optimal for Q2. Hence a clustered index on value would be optimal.`

27. **6a) Hardware and DBMS Design**

MULTI | 5 points | 0.10 penalty | Multiple

6a) Select the true statements:

(a) In a main-memory relational database system, it remains necessary to log the "old" values during updates, in case the transaction aborts. (50%)
(b) NoSQL literally means that SQL-like query languages must not be supported by NoSQL systems. (−50%)
(c) Document stores cannot store business data that is typically stored in relational systems. (−50%)
(d) While locking whole tables guarantees serializability in a relational system, it leads to terrible performance, which is why it is turned off by default in most systems. (50%)

28. **6b) Hardware and DBMS Design**

ESSAY | 5 points | 0.10 penalty | monospaced

6b) Write your reflections here:

Notes for grader:

- `Pros: Reduces likelihood of losing data; Increases availability in case of network partitions. Cons: Increases the work required to maintain consistency, especially if strict consistency is desired; Increases storage requirements.`

29. **7a) Data Systems for Analytics**

MULTI | 5 points | 0.10 penalty | Multiple

7a) Select the true statements:

(a) All big data applications are evil in nature. (−50%)
(b) Spark is designed for supporting user-facing and interactive applications. (−50%)
(c) Many big data applications require so much computation that a distributed processing framework is necessary. (100%)
(d) A machine learning model that is based on real data can never have any bias, because it accurately models real life. (−50%)

30. **7b) Data Systems for Analytics**

ESSAY | 5 points | 0.10 penalty | monospaced

7b) Write your reflections here:

Notes for grader:

- The most important access pattern is sequential IO. Adding ‘‘ over large collections’’ would be even more accurate. This is because the typical big data applications aim to model the entirety of the data somehow, and to do that the whole data must be read. Furthermore, many approaches require repeatedly reading the collection. However, the specific order of reading the collection is rarely important, allowing for sequential reads in whatever order the collections happen to be in.

Total of marks: 105