Introduction to Database Design / Data Management BSc Exam

Björn Thór Jónsson

April 28, 2020

Instructions

You have 4 hours to answer the 6 problems described in the following 9 numbered pages.

Instructions for SQL Queries in Question 1

Queries must work for any database instance and should avoid system-specific language 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 of the problem with the query, and it may be given partial points.

Database Description for Questions 1–2

In this exam you will work with a fictional database of employees and departments. To start working with the database, run the commands in `idb-april-2020-DB.sql` found in LearnIT using the PostgreSQL DBMS on your laptop. It is recommended to use `psql` for this purpose, especially since the largest table contains about 1 million rows.

The database contains a variety of information in the following schema:

> *-- entity tables*
> *employees(emp_no, birthday, name)*
> *departments(dept_no, name)*
>
> *-- relationship tables*
> *dept_emp(emp_no, dept_no, from_date, to_date)*
> *dept_manager(emp_no, dept_no, from_date, to_date)*
> *salaries(emp_no, salary, from_date, to_date)*
> *titles(emp_no, title, from_date, to_date)*

Attributes are self-explanatory. Primary keys are correctly defined, as underlined above, and foreign keys are also correctly defined. You may study the DDL commands to understand the details of the tables (the CREATE TABLE statements are at the top of the script) or inspect the tables using SQL queries. Two additional notes are in order:

- There are multiple attributes of type *date* in the database. Two dates can be compared with the regular operators (=, <, >, ...) and the length of their time span can be computed using the regular minus (–) operator. The `extract` function can be used to extract parts of dates: you may use `extract(year from birthday)` to get the birthyear of an employee, to give an example.
- A `null` value in the *to_date* attribute is used to denote *current* status in the four relationship tables. To find out who is the current manager of all departments, for example, the following query can be used:

  ```
  select *
  from dept_manager
  where to_date is null;
  ```

  Similar queries can be used to find the current department, title, and salary, of an employee.

1 SQL (40 points)

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

(a) There are 22913 employees who were born in 1964. How many employees were born in 1954?

(b) In which year was the current manager of the 'Finance' department born?

Note: This query returns a year, not a count.

(c) There are 3 departments who have had more than 50,000 employees in total. How many departments have more than 50,000 *current* employees?

(d) Employee number 429386 is the *former* employee with the longest time-span in a single department (from '1985-02-03' to '2002-08-01'). What is the employee number of the former employee with the longest *total* time-span across all departments?

Note: This query must ignore employees that are currently employed. Also, this query returns an emp_no, not a count.

(e) In the 'Production' department, there have been in total 8390 employees who have never held any title ending with 'Engineer'. How many such employees have been in total in the 'Development' department?

(f) How many different titles do the current employees of the 'Development' department hold?

(g) There are 66263 employees who have held all titles ending with 'Staff'. How many employees have held all titles ending with 'Engineer'?

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

(h) Employee number 63966 is the current 'Senior Engineer' in the 'Development' department who has the lowest salary of all such employees. What is the employee number of the current 'Senior Engineer' in the 'Development' department who has the *highest* salary of all such employees.

Note: This query returns an emp_no, not a count.

Enter each query, along with its numerical answer, in LearnIT. Queries must adhere to the detailed guidelines given on Page 1.

2 (BSc ONLY) SQL programming (5 points)

Consider the SQL trigger code in Figure 1.

```
-- Trigger function
CREATE FUNCTION CheckDeptEmp() RETURNS TRIGGER
AS $$ BEGIN
  -- Check 1: Is the hiring time range OK?
  IF (NEW.to_date < NEW.from_date) THEN
    RAISE EXCEPTION 'Cannot be hired for a negative duration'
    USING ERRCODE = '45000';
  END IF;
  -- Check 2: Is the employee currently hired?
  IF (NEW.to_date IS NULL AND
      EXISTS (SELECT *
              FROM dept_emp DE
              WHERE DE.emp_no = NEW.emp_no
                AND DE.to_date IS NULL)) THEN
    RAISE EXCEPTION 'Cannot be currently employed in two departments'
    USING ERRCODE = '45000';
  END IF;
  RETURN NEW;
END; $$ LANGUAGE plpgsql;

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

-- Test the trigger
INSERT INTO dept_emp VALUES (10010, 'd007', '2000-06-26', NULL);
INSERT INTO dept_emp VALUES (10010, 'd006', '2000-06-26', '2000-06-26');
INSERT INTO dept_emp VALUES (10010, 'd007', '2000-06-26', '2000-06-01');
```

Figure 1: Insertion trigger `CheckDeptEmp` for the `dept_emp` relation.

a) Select the true statements:

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

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

b) Based on the `dept_emp` relation instance installed by the database installation script (you can use queries to check the relation instance for employee 10010), which INSERT statements will give an error (note that more than one of the statements may give an error):

(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.

3 ER Diagrams and Normalization (25 points)

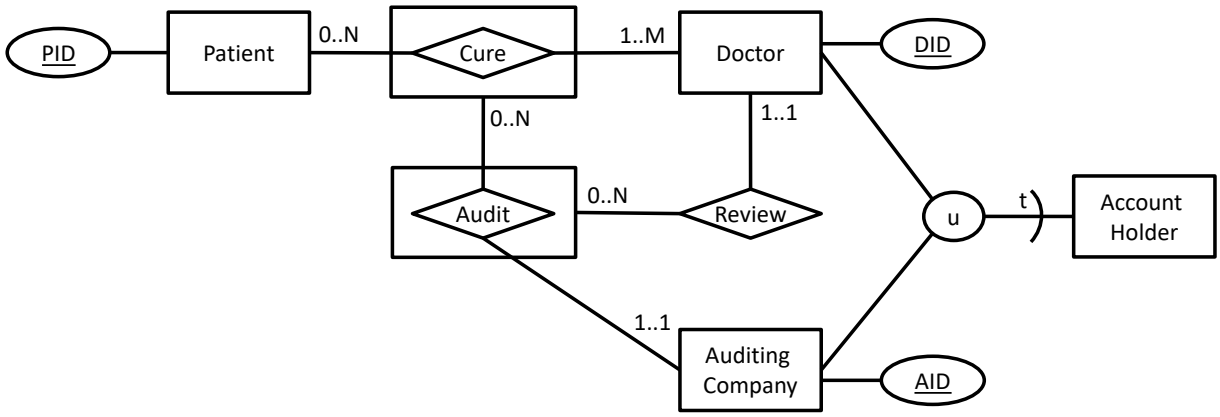

Figure 2: ER Diagram for a medical facility database.

a) The ER diagram in Figure 2 shows a database for a medical facility. Select the true statements. You should base your answers **only** on the ER diagram:

(a) Every cure is reviewed by exactly one doctor via an audit.

(b) A doctor may review his/her own cure via an audit.

(c) Every doctor reviews at least one audit.

(d) Each doctor is connected to at least one auditing company via the Cure and Audit relationships.

(e) Each patient is connected to exactly one auditing company via the Cure and Audit relationships.

(f) An account holder can be both a doctor and an auditing company.

b) Write SQL DDL commands to create a medical facility database based on the ER diagram in Figure 2 using the methodology given in the textbook and lectures. The DDL script *must run* in PostgreSQL. The relations must include all relevant primary key, candidate key, foreign key and NOT NULL constraints. Constraints that cannot be enforced with these standard constraints should be omitted. All attributes should be of type INTEGER or SERIAL.

c) Write an ER diagram for a drug database. The diagram should clearly show the entities, relationships and participation constrains described below. Attributes are only important if they are mentioned in this description; you should not add other attributes. Follow precisely the notation presented in the textbook and lectures.

- A drug has a unique generic name.
- Drugs may be produced by multiple manufacturers, and each manufacturer may produce the same drug using multiple different brand names.
- A chemical is either an atom or a compound, not both.
- A drug is composed of at least one chemical.
- Suppliers supply chemicals for the production of drugs, with these constraints:
  - Each brand-name drug production uses exactly one supplier for each chemical.
  - We must ensure that the chemicals used for the brand-name production are actually required for the drug.

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

$$\begin{aligned} A &\rightarrow CDE \\ B &\rightarrow E \\ B &\rightarrow A \\ CD &\rightarrow E \end{aligned}$$

Select the true statements:

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

(b) $A \rightarrow C$ is an unavoidable functional dependency.

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

(d) The relation can be normalized to BCNF without losing functional dependencies (excluding trivial, unavoidable, and derivable dependencies) .

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

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

Normalize R to the highest possible normal form based on functional dependencies (3NF or BCNF), while allowing all functional dependencies (excluding trivial, unavoidable, and derivable dependencies) to be checked within a single relation. Write down the resulting relations and clearly indicate whether they are in BCNF.

4 Index Selection (10 points)

Consider the following large relation with information on sales:

Sales(saleID, prodID, repID, date, quantity, price)

For each of the queries below, select the index that a good query optimiser is most likely to use for the Sales table to process the query. Assume that all indexes are *unclustered* B+-trees. Also, assume that the relation has sales data for the last 30 years, that each day has roughly the same number of sales, and that there are 100 prodIDs and 100 repIDs. For each case, select the best index for the query, or select "no index" if a full table scan would yield better performance than any of the possible indexes.

(a) Sales(saleID)

(b) Sales(prodID)

(c) Sales(repID)

(d) Sales(repID, price)

(e) Sales(date)

(f) No index

The queries are:

Query 1

```
select *
from Sales
where date = '12-12-2010';
```

Query 2

```
select sum(price)
from Sales
where repID = 100;
```

Query 3

```
select avg(date)
from Sales;
```

Query 4

```
select avg(date)
from Sales
where quantity > 100;
```

5 Hardware and DBMS Design (10 points)

a) Select the correct statements below:

(a) Storing multiple database recovery logs on a single hard disk drive (HDD) gives excellent logging performance.

(b) Data replication in a distributed system reduces the risk of losing data.

(c) The notion of *consistency* in the CAP theorem is the same as the notion of *consistency* in the ACID properties.

(d) Compared to hard-disk drive (HDD) technology, state-of-the-art solid state disks (SSDs) generally improve the performance of disk operations.

b) Social media providers, such as Facebook, use *multiple copies* of data and *eventual consistency* for most of their functionality. As an example, comments on posts may not be seen immediately, or even in the same order, by all users. Discuss briefly how performance would be affected if *strong consistency* were adopted instead. Base your discussion on a concrete example, such as the timing and order of comments described above.

6 Data Systems for Analytics (10 points)

a) Select the correct statements below:

(a) Spark is more efficient than Hadoop for most Big Data applications.

(b) Videos are generally considered *structured data*.

(c) Spark has very low latency for small tasks.

(d) Distributed processing is important for most Big Data applications.

b) Discuss the meaning of *Variety* in Big Data applications.

Retake Exam April 2020

1. **0) Declaration**

   0) Upload your declaration of independent work.

   Notes: (not included in XML)

   - `Not included`

2. **1a) SQL**

   1a) Write your SQL query here:

   Notes: (not included in XML)

   - `See idb-april-2020-SQL.sql`

3. **1a) Numerical answer**

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

   - 23228 ✓

4. **1b) SQL**

   1b) Write your SQL query here:

   Notes: (not included in XML)

   - `See idb-april-2020-SQL.sql`

5. **1b) Numerical answer**

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

   - 1957 ✓

6. **1c) SQL**

   1c) Write your SQL query here:

   Notes: (not included in XML)

   - `See idb-april-2020-SQL.sql`

7. **1c) Numerical answer**

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

   - 2 ✓

8. **1d) SQL**

   1d) Write your SQL query here:

   Notes: (not included in XML)

- `See idb-april-2020-SQL.sql`

9. **1d) Numerical answer**

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

   - 300023 ✓

10. **1e) SQL**

    1e) Write your SQL query here:

    Notes: (not included in XML)

    - `See idb-april-2020-SQL.sql`

11. **1e) Numerical answer**

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

    - 9456 ✓

12. **1f) SQL**

    1f) Write your SQL query here:

    Notes: (not included in XML)

    - `See idb-april-2020-SQL.sql`

13. **1f) Numerical answer**

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

    - 7 ✓

14. **1g) SQL**

    1g) Write your SQL query here:

    Notes: (not included in XML)

    - `See idb-april-2020-SQL.sql`

15. **1g) Numerical answer**

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

    - 3008 ✓

16. **1h) SQL**

    1h) Write your SQL query here:

    Notes: (not included in XML)

- See `idb-april-2020-SQL.sql`

17. **1h) Numerical answer**

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

    - 86631 ✓

18. **2a) SQL programming**

    2a) Select the true statements:

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

19. **2b) SQL programming**

    2b) Select the true statements:

    (a) The first INSERT statement will give an error. (33.33333%)
    (b) The second INSERT statement will give an error. (33.33333%)
    (c) The third INSERT statement will give an error. (33.33333%)

20. **3a) ER Diagram Interpretation**

    3a) Select the true statements:

    (a) Every cure is reviewed by exactly one doctor via an audit. (50%)
    (b) A doctor may review his/her own cure via an audit. (50%)
    (c) Every doctor reviews at least one audit. (0%)
    (d) Each doctor is connected to at least one auditing company via the Cure and Audit relationships. (0%)
    (e) Each patient is connected to exactly one auditing company via the Cure and Audit relationships. (0%)
    (f) An account holder can be both a doctor and an auditing company. (0%)

21. **3b) SQL DDL**

    3b) 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 `idb-april-2020-DDL.sql`

22. **3c) ER Diagram Creation**

    3c) Upload the ER diagram or deliver a hand drawing at the exam.

    Notes: (not included in XML)

- `See idb-april-2020-ER.jpg`

23. **3d) Normalisation**

    3d) Select the true statements:

    (a) AB is the only (candidate) key of R. (0%)
    (b) $A \rightarrow C$ is an unavoidable functional dependency. (0%)
    (c) Normalizing to 3NF or BCNF results in exactly two relations. (0%)
    (d) The relation can be normalized to BCNF without losing functional dependencies (excluding trivial, unavoidable, and derivable dependencies) . (100%)

24. **3e) Normalisation**

    3e) Write down the normalized relations.

    Notes: (not included in XML)

    - `R1(A, B, C) - not in BCNF; R2(B, E) - in BCNF; R3(E, D) - in BCNF`

25. **4a) Query 1**

    4a) Selection for Query 1:

    (a) Sales(saleID)
    (b) Sales(prodID)
    (c) Sales(repID)
    (d) Sales(repID, price)
    (e) Sales(date) ✓
    (f) No index

26. **4b) Query 2**

    4b) Selection for Query 2:

    (a) Sales(saleID)
    (b) Sales(prodID)
    (c) Sales(repID)
    (d) Sales(repID, price) ✓
    (e) Sales(date)
    (f) No index

27. **4c) Query 3**

    4c) Selection for Query 3:

    (a) Sales(saleID)
    (b) Sales(prodID)
    (c) Sales(repID)
    (d) Sales(repID, price)

(e) Sales(date) ✓
(f) No index

28. **4d) Query 4**

4d) Selection for Query 4:

(a) Sales(saleID)
(b) Sales(prodID)
(c) Sales(repID)
(d) Sales(repID, price)
(e) Sales(date)
(f) No index ✓

29. **5a) Hardware and DBMS Design**

5a) Select the true statements:

(a) Storing multiple database recovery logs on a single hard disk drive (HDD) gives excellent logging performance. (0%)
(b) Data replication in a distributed system reduces the risk of losing data. (50%)
(c) The notion of *consistency* in the CAP theorem is the same as the notion of *consistency* in the ACID properties. (0%)
(d) Compared to hard-disk drive (HDD) technology, state-of-the-art solid state disks (SSDs) generally improve the performance of disk operations. (50%)

30. **5b) Hardware and DBMS Design**

5b) Write your reflections here:

Notes: (not included in XML)

- `There are two ways to move to strong consistency. a) A single copy is stored, and all updates happen to that copy. This would mean serious bottlenecks of access, as all accesses and updates to popular material would go to the same server. b) All copies are kept up to date. This would mean that updates and/or reads must access many copies simultaneously, which also would result in serious bottlenecks of access.`

31. **6a) Data Systems for Analytics**

6a) Select the true statements:

(a) Spark is more efficient than Hadoop for most Big Data applications. (50%)
(b) Videos are generally considered *structured data*. (0%)
(c) Spark has very low latency for small tasks. (0%)
(d) Distributed processing is important for most Big Data applications. (50%)

32. **6b) Data Systems for Analytics**

6b) Write your reflections here:

Notes: (not included in XML)

- `Variety means that the data can a) be from various origins, b) address various aspects of the big data archive, and c) have various formats.`