Practice Exam

Databases

Pierre Senellart

10 June 2026

You have 2 hours to answer this practice exam. You are allowed 10 A4 sheets (i.e., 20 pages) with the content of your choice. No electronic devices or any other material is allowed. Do not forget to add your name on all your answer sheets and to number them.

The exam consists of seven parts (A–G), which are largely independent; it is graded out of 20 points. The number of points is indicated for each question. Write readable SQL: minor syntactic slips are tolerated as long as the intent is unambiguous.

VéloCampus is a campus bike-sharing service. Members borrow bikes from docking stations, ride them, and return them at a (possibly different) station. The service wants a relational database to manage its members, stations, bikes, and trips. The domain is described as follows:

- A *member* has a unique id, a name, an email, a subscription plan, and a signup date.
- A *station* has a unique id, a name, a geographic location (latitude and longitude), and a capacity (number of docking points).
- A *bike* has a unique id, a model, and a current status. Every bike is registered at exactly one *home* station; a station is the home of many bikes.
- A *trip* records that a member borrowed a bike at a start station and returned it at an end station, with a start time and an end time. A member can make many trips; each trip concerns exactly one member, one bike, one start station, and one end station.

A. Modelling and Design (3.5 points)

1. (2 points) Draw an Entity–Relationship diagram for this domain. Show the entity sets, their attributes (underline the key of each entity set), the relationships, and the cardinality constraints.
2. (1 point) Translate your ER diagram into a relational schema. For each relation, underline the primary key and indicate the foreign keys.
3. (0.5 point) Give one integrity constraint of this domain that *cannot* be enforced by keys and foreign keys alone, and say briefly how a DBMS could still enforce it.

Reference schema for the remaining parts. Regardless of your answer to Part A, use the following schema from Part B onwards:

- `member(id, name, email, plan, signup_date)`
- `station(id, name, latitude, longitude, capacity)`
- `bike(id, model, status, home_station)`
- `trip(id, member, bike, start_station, end_station, start_time, end_time)`

B. Relational Algebra (2 points)

Write a relational-algebra expression for each of the following queries.

4. (0.5 point) The names of stations with a capacity of at least 30.
5. (1.5 points) The models of bikes that have never been used in any trip.

C. Relational Calculus (1 point)

Use *domain* relational calculus with positional predicates, where the attribute order is the one of the reference schema, for instance `member`(i, n, e, p, d) and `trip`$(i, m, b, ss, es, st, et)$.

6. (1 point) The names of stations that are the start station of at least one trip made by a member on the `'student'` plan.

D. SQL (6 points)

Write an SQL query for each of the following.

7. (1.5 points) The names and emails of all members on the `'student'` plan.
8. (1.5 points) The email of every member who has taken at least 5 trips.
9. (1.5 points) The 10 stations from which the most trips have started, together with that number of trips, most popular first.
10. (1.5 points) The names of members who have *never* returned a bike at the station named `'Bastille'`. Use `NOT EXISTS`.

E. Functional Dependencies and Normalization (3 points)

To bill rides, an intern proposes a single relation

$$\texttt{RIDE}(\underline{\texttt{trip}}, \texttt{member}, \texttt{email}, \texttt{plan}, \texttt{rate}, \texttt{minutes})$$

whose only key is `trip`, with the functional dependencies

$$\texttt{trip} \to \texttt{member,minutes} \qquad \texttt{member} \to \texttt{email,plan} \qquad \texttt{plan} \to \texttt{rate}.$$

11. (1 point) Justify that `RIDE` is in 2NF, and exhibit a functional dependency that prevents it from being in 3NF.

12. (1.5 points) Give a decomposition of `RIDE` into BCNF.

13. (0.5 point) Is your decomposition dependency-preserving? Justify briefly.

F. Storage, Indexing, and Query Optimization (2.5 points)

14. (1 point) The `trip` table has 10^7 rows. Two queries are very frequent: (a) "all trips of a given member", and (b) "all trips that started during a given day". Which index(es) would you create, and of what kind?

15. (1.5 points) Consider the algebra expression

$$\sigma_{station.name=\text{'Bastille'}}\left(\texttt{trip} \bowtie_{start\_station=id} \texttt{station}\right).$$

Give an equivalent but cheaper-to-evaluate expression, name the equivalence rule you used, and explain in one sentence why it is faster.

G. Databases from Python and the Web (2 points)

16. (1 point) The following Python (DB-API 2.0) looks up a member by id, where `mid` comes from user input:

```
cur.execute("SELECT name, email FROM member WHERE id = " + mid)
```

(a) Rewrite it as a *parameterized* query. (b) Why does this matter?

17. (1 point) The service wants a small search box on its home page. Write the HTML for a form that lets a visitor search stations by name: it must send a `GET` request to `/stations`, with a single text field named `q` and a submit button.