

4 Database Performance (Tutorial 9)

Started: 10 Nov at 21:00

Quiz instructions

Purpose

Databases have to be responsive to be useful. This exercise helps you decide where indexes can help and whether an index is useful.

This is an individual assessment worth 5% of your final mark.

Unit learning outcomes

1. Apply a design thinking approach to understand and solve a stakeholder problem.
3. Apply ethical, professional and technical considerations in the development of the project (data management) solution.
5. Identify project-related skills requirements, locate suitable resources and acquire the appropriate skills mostly independently.

Graduate Attributes

- Communication 1 - Verbal communication
- Digital literacies 1 – Information literacy
- Digital Literacies 2 – Technical literacy

Submission Requirements

These activities should be completed as **part of tutorial 9**. You are **required to come to the tutorial to answer potential questions** about it. The tutor **may not mark it** in your absence. The submission is due after the tutorial of week 10, to give you extra time to complete the activities if needed.

Question 1

1 pts

The company has frequent look-ups of its employees by ID, i.e. whenever a person's job or salary is reviewed on a screen. The query is:

```
SELECT * FROM employees WHERE emp_no = 212900;
```

Would you consider adding an index to the employees table to make this query faster? Discuss and give reasons.

Adding index to employees table may make this query faster. However, the **benefits of an index in this case may be minimal. Indexes are most effective when there is a variety of queries that can benefit from them.** So, in this case, the improvement in the query speed might not be noticeable. Moreover, since emp_no is **primary key**, **MySQL automatically creates an index on it so adding another index might be redundant.** In the future, the table may expand (currently it is small) so the **presence of indexes can add complexity and maintenance overhead.** So, I **wouldn't consider** adding an index to the employees table to make this query faster.

p ▶ strong



110 words



Question 2

1 pts

To determine the eligibility of employees for certain bonuses, the employees often have to be listed by age (or rather, birth year).

```
SELECT emp_no FROM employees WHERE YEAR(birth_date) <= 1960;
```

What could potentially be indexed in this case? Add the index and investigate whether it is used when you run this query. What do you conclude?

birth_date is potentially indexed in this case to improve the performance of the query. The query to add index in this case is :
"CREATE INDEX idx_birth_date ON employees (birth_date);" This index will speed up the search process for employees whose birth year is less than or equal to 1960. Moreover, **using YEAR function may prevent the MySQL database from using the index effectively.** So, instead **we should specify the date and month: "SELECT emp_no FROM employees WHERE birth_date <= '1960-12-31';"** I conclude that the use of index can potentially improve the performance of the query in this case, but the **actual effectiveness depends on**

various factors, including the size of the table and the distribution of data.

p ▶ strong



117 words



Question 3

1.5 pts

Sometimes the company has to calculate leave entitlements, and for this it is helpful to list employees by hire date:

```
SELECT emp_no, first_name, last_name FROM employees ORDER BY hire_date;
```

Add an index where you think it would best help speed up this query and see if it is used.

hire_date should be indexed to boost the query speed. The query for creating the index for the column hire_date is: **"CREATE INDEX idx_hire_date ON employees (hire_date);"**. This index will **improve the performance** of the clause **"ORDER By hire_date"**. There is a way to check whenever the index is used in this case or not by using **EXPLAIN**: **"EXPLAIN SELECT emp_no, first_name, last_name FROM employees ORDER BY hire_date;"**. **Adding the index on the hire_date column can significantly speed up sorting operations, making the calculation more efficient**, especially when the **table contains a large number of records**.

p



95 words



Question 4








1.5 pts

Sometimes the company wants to determine if there is gender bias in appointments to higher positions. For this purpose, male and female employees have to be fetched separately:

```
SELECT emp_no, first_name, last_name FROM employees WHERE gender = 'F';
```

Add an index where you think it would best help speed up this query and see if it is used.

Edit View Insert Format Tools Table

12pt ▾ Paragraph ▾ | **B** *I* U A ▾  ▾ T^2 ▾ |  ▾  ▾  ▾  ▾ |  **H-P**  ▾ | ⋮

In this case, **gender column** will be indexed with the query: "**CREATE INDEX idx_gender ON employees (gender);**". It will **boost the performance** "**WHERE gender = 'F'**" condition. After creating the index, I can check with the query: "**EXPLAIN SELECT emp_no, first_name, last_name FROM employees WHERE gender = 'F';**" then if the Using index condition is displayed, the **index is being utilized for the query**. Adding the index on the gender column can speed of the retrieval of female employees in table contains a large number of records. This can be **useful for analyzing gender distribution in appointments to higher positions**.

p ▶ strong



100 words



Saved at 21:49

Submit quiz