Web Programming

Databases part 1 queries

Leander Jehl | University of Stavanger

Learning goal:

- Advanced queries
- ORDER BY
- WHERE with LIKE, AND, OR
- GROUP BY
- JOIN

Examples at

github.com/dat310-2025/info/tree/master/
exercises/sql/query

ORDER BY

SELECT

- Select named columns

```
SELECT ID, name FROM department;
```

- Select all columns

```
SELECT * FROM department;
```

- Select rows with specific values

```
SELECT name FROM department WHERE ID = 0;
```

ORDER BY

- Order results by one column

```
SELECT ID, name FROM department ORDER BY name;
```

- Order by column need not be selected

```
SELECT ID, name FROM employee ORDER BY departmentId;
```

- Can order by multiple columns

```
SELECT name, departmentID FROM employee ORDER BY departmentId, name;
```

- Order by comes after WHERE

```
SELECT name FROM employee WHERE departmentID = 0 ORDER BY name;
```

```
SELECT name FROM employee ORDER BY na[illegible] WHERE departmentID = 0;
```

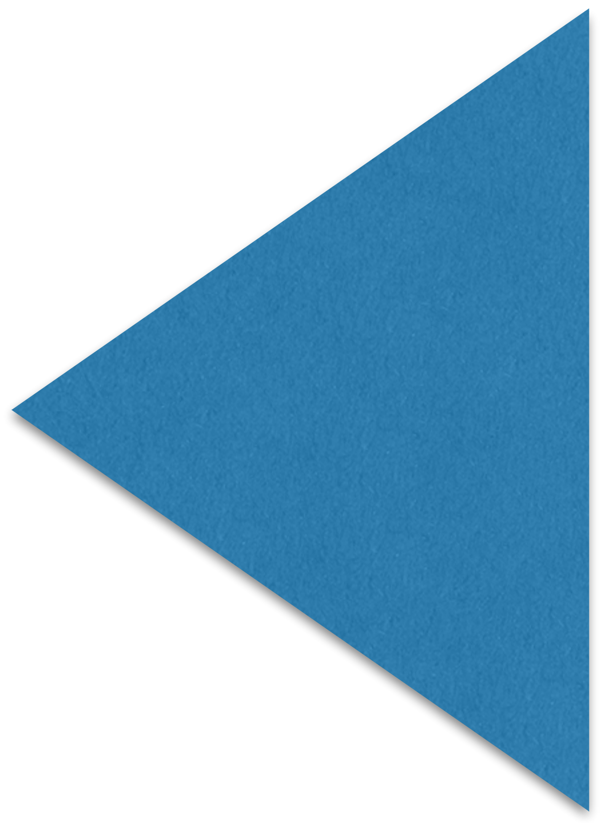
WHERE

WHERE

- **WHERE** is used to only select some rows

```
SELECT name FROM employee WHERE departmentID = 1;
```

Name
“Tom"
“Ida”



Employee

ID	Name	Salary	Office	Deptment
1234	“Tom"	50k	E123	1
1235	”Bjørn”	?	E245	2
1345	“Ida”	60k		1

LIKE

- **LIKE** matches strings not case sensitive

```
SELECT name FROM employee WHERE name LIKE ‘tom’;
```

- Add **%** for any number of arbitrary signs
  - Names that start with letter e:

    ```
    SELECT name FROM employee WHERE name LIKE ‘e%’;
    ```

  - Names that contain letter e:

    ```
    SELECT name FROM employee WHERE name LIKE ‘%e%’;
    ```

AND/OR

- **AND** combines multiple conditions inside **WHERE**:

```
SELECT name FROM employee WHERE name LIKE ‘tom’ AND departmentID = 1;
```

- **OR** looks for one of two conditions

```
SELECT name FROM employee WHERE name LIKE ‘%e%’ OR room LIKE ‘%e%’;
```

NULL

- Check for missing values using **IS NULL**

```
SELECT ID FROM employee WHERE salary IS NULL;
```

```
SELECT ID FROM employee WHERE salary IS NOT NULL;
```

Exercises #1

github.com/dat310-2025/info/tree/master/
exercises/sql/query

GROUP BY

GROUP BY

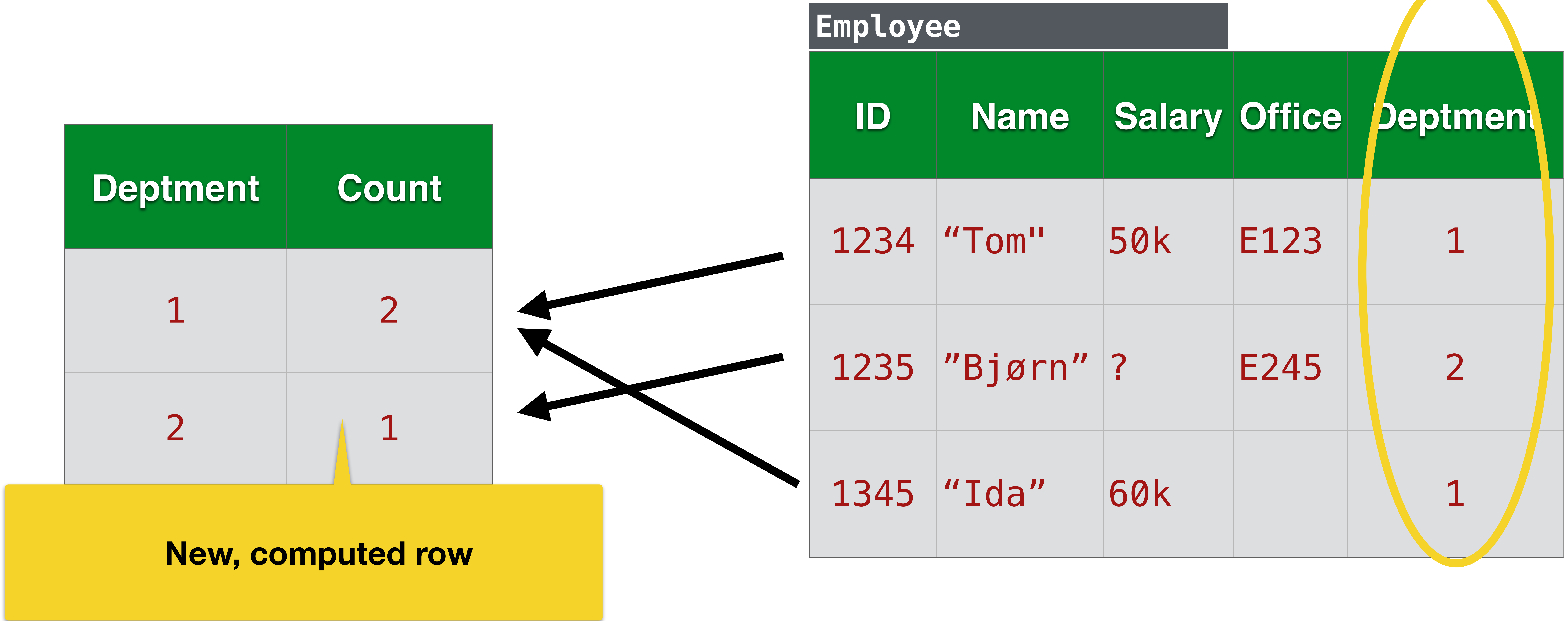
- Combine multiple rows with the same value in one column

Employee

ID	Name	Salary	Office	Deptment
1234	“Tom"	50k	E123	1
1235	”Bjørn”	?	E245	2
1345	“Ida”	60k		1

GROUP BY

- Combine multiple rows with the same value in one column



Deptment	Count
1	2
2	1

Employee

ID	Name	Salary	Office	Deptment
1234	“Tom"	50k	E123	1
1235	”Bjørn”	?	E245	2
1345	“Ida”	60k		1

GROUP BY

- Combine multiple rows with the same value in one column
- Example:

```
SELECT departmentID, COUNT(*) AS count FROM employee GROUP BY departmentID;
```

- `AS count` assigns a column name.
  This is not mandatory but highly recommendet.

GROUP BY

- In group by, we often want to compute the row values

```
SELECT departmentID, COUNT(*) AS count FROM employee GROUP BY departmentID;
```

- We can use the following functions:
  - COUNT(*)
  - MAX(column name)
  - AVG(column name)
  - SUM(coumn name)

GROUP BY

- Select offices seating more than 2:

```
SELECT office, COUNT(*) AS count
FROM employee
    GROUP BY office;
```

- Can not have WHERE after GROUP BY

```
SELECT office, CO[illegible](*)[illegible] count
FROM employee
    GROUP BY offic[illegible]
    WHERE count >[illegible]AN[illegible]fice IS NOT NULL;
```

GROUP BY

- Select offices seating more than 2:

```
SELECT office, COUNT(*) AS count
FROM employee
    GROUP BY office;
```

- Use HAVING after GROUP BY to filter results

```
SELECT office, COUNT(*) AS count
FROM employee
    GROUP BY office
    HAVING count > 2 AND office IS NOT NULL;
```

JOIN

JOIN

- Combine information from multiple tables.
- E.g. show employees with department name.

Employee

ID	Name	Salary	Office	Deptment
1234	“Tom"	50k	E123	1
1235	”Bjørn”	?	E245	2
1345	“Ida”	60k		1

Department

ID	Name
1	“Engineering”
2	“HR”
3	“Engineering 2”

1234	“Tom"	50k	E123	1
1235	”Bjørn”	?	E245	2
1345	“Ida”	60k		1

Employee

ID	Name
1	“Engineering”
2	“HR”
3	“Engineering 2”

Department

ID	Name	Salary	Office	Deptment	Name
1234	“Tom"	50k	E123	1	“Engineering”
1235	”Bjørn”	?	E245	2	“HR”
1345	“Ida”	60k		1	“Engineering”

1234	“Tom"	50k	E123	1
1235	”Bjørn”	?	E245	2
1345	“Ida”	60k		1

Employee

ID	Name
1	“Engineering”
2	“HR”
3	“Engineering 2”

Department

ID	Name	Salary	Office	Deptment	Name
1234	“Tom"	50k	E123	1	“Engineering”
1235	”Bjørn”	?	E245	2	“HR”
1345	“Ida”	60k		1	“Engineering”

JOIN

- Combine information from multiple tables.
- E.g. show employees with department name.

```
SELECT employee.name, department.name AS department
FROM employee JOIN department
    ON employee.departmentID = department.ID;
```

JOIN

- Combine information from multiple tables.
- E.g. show employees with department name.

```
SELECT employee.name, department.name AS department
FROM employee JOIN department
    ON employee.departmentID = department.ID;
```

Tell on which column to join!

JOIN

- Combine information from multiple tables.
- E.g. show employees with department name.

```
SELECT employee.name, department.name AS department
FROM employee JOIN department
    ON employee.departmentID = department.ID;
```

Use tablename.columnname to access one column.

JOIN

- Combine information from multiple tables.
- E.g. show employees with department name.

```
SELECT employee.name, department.name AS department
FROM employee JOIN department
    ON employee.departmentID = d
```

Rename ambigous columns.

JOIN

- Combine information from multiple tables.
- E.g. show employees with department name.

```
SELECT employee.name, department.name AS department
FROM employee JOIN department
    ON employee.departmentID = d
```

Rename ambigous columns.

JOIN

- Combine information from multiple tables.
- E.g. show employees with department name.

```
SELECT employee.name, department.name AS department
FROM employee JOIN department
    ON employee.departmentID = department.ID;
```

- JOIN is also called INNER JOIN

Rows without match or with NULL are excluded from JOIN results.

1234	“Tom"	50k	E123	1
1235	”Bjørn”	?	E245	2
1345	“Ida”	60k		NULL

Employee

ID	Name
1	“Engineering”
2	“HR”
3	“Engineering 2”

Department

NULL row removed from JOIN

ID	Name	Salary	Office	Deptment	Name
1234	“Tom"	50k	E123	1	“Engineering”
1235	”Bjørn”	?	E245	2	“HR”

LEFT JOIN

- Combine information from multiple tables.
- E.g. show employees with department name.

```
SELECT employee.name, department.name AS department
FROM employee LEFT JOIN department
    ON employee.departmentID = department.ID;
```

Employee left from join

- LEFT JOIN includes all rows from the left table

LEFT JOIN

- Combine information from multiple tables.
- E.g. show employees with department name.

```
SELECT employee.name, department.name AS department
FROM employee LEFT JOIN department
    ON employee.departmentID = department.ID;
```

Employee left from join

- LEFT JOIN includes all rows from the left table

Rows without match or with NULL are included in JOIN results.

1234	“Tom"	50k	E123	1
1235	”Bjørn”	?	E245	2
1345	“Ida”	60k		NULL

Employee

ID	Name
1	“Engineering”
2	“HR”
3	“Engineering 2”

Department

Included in **LEFT JOIN**

ID	Name	Salary	Office	Deptment	Name
1234	“Tom"	50k	E123	1	“Engineering”
1235	”Bjørn”	?	E245	2	“HR”
1345	“Ida”	60k		NULL	NULL

EXAMPLE

- Find employees that share a room:

Need AS with alias
to tell two tables apart.

```
SELECT e1.name, e2.name FROM employee AS e1 JOIN
employee AS e2
    ON e1.office = e2.office;
```

EXAMPLE

- Find employees that share a room:

Need AS with alias to tell two tables apart.

```
SELECT e1.name, e2.name FROM employee AS e1 JOIN
employee AS e2
    ON e1.office = e2.office;
```

Gives more rows than employees!

EXAMPLE

- Find employees that share a room:

```
SELECT e1.name, e2.name
FROM employee AS e1 JOIN employee AS e2
    ON e1.office = e2.office
    WHERE e1.ID != e2.ID;
```

- Or

-
```
SELECT e1.name, e2.name
FROM employee AS e1 JOIN employee AS e2
    ON (e1.office = e2.office
        AND e1.ID != e2.ID);
```

Nested Queries

Nested Queries

- Queries can be nested inside each other.
- Place nested query inside brackets (…)
- E.g. JOIN with the result of a query
- WHERE row IN (QUERY)

```
SELECT name from employee
WHERE office IN (
    SELECT office, count(ID) AS count
    FROM employee
    GROUP BY office
    HAVING count > 1 AND office IS NOT NULL
);
```

Exercises #2

github.com/dat310-2025/info/tree/master/
exercises/sql/query