Database Modeling and Database Systems — Unit 1

Dr.-Ing Anna Androvitsanea

IU Internationale Hochschule GmbH

October 30, 2023

TOPIC OUTLINE

Relational Database Basics	1
Database Queries to Exactly One Table	2
Conception and Modeling of Relational Databases	3
Creating Relational Databases	4
Complex Database Queries on Multiple Tables	5

TOPIC OUTLINE

Manipulating Records in Databases	**6**
NoSQL Database System	**7**

Study goals

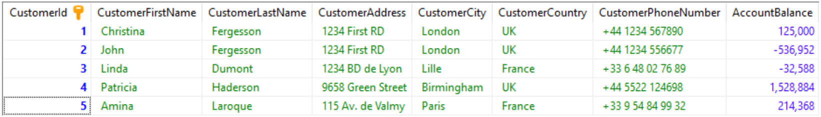
- Use the SQL SELECT statement to query data in an Relational Database Management Syste (RDBMS).
- Filter query results using the WHERE clause.
- Group query results using the GROUP BY and HAVING clauses.
- Explain and use subqueries.

EXPLAIN SIMPLY

1. What is the difference between data and information?
2. Why would you use queries instead of checking the entire contents of a table (2 reasons)?
3. What is the general form of an SQL SELECT statement?

Let's start with the study goals

- Use the SQL SELECT statement to query data in an Relational Database Management Syste (RDBMS).
- Filter query results using the WHERE clause.
- Group query results using the GROUP BY and HAVING clauses.
- Explain and use subqueries.

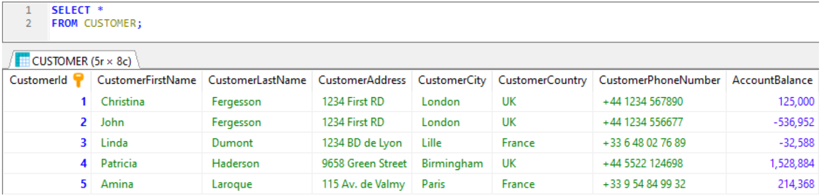


CustomerId 🔑	CustomerFirstName	CustomerLastName	CustomerAddress	CustomerCity	CustomerCountry	CustomerPhoneNumber	AccountBalance
1	Christina	Fergesson	1234 First RD	London	UK	+44 1234 567890	125,000
2	John	Fergesson	1234 First RD	London	UK	+44 1234 556677	-536,952
3	Linda	Dumont	1234 BD de Lyon	Lille	France	+33 6 48 02 76 89	-32,588
4	Patricia	Haderson	9658 Green Street	Birmingham	UK	+44 5522 124698	1,528,884
5	Amina	Laroque	115 Av. de Valmy	Paris	France	+33 9 54 84 99 32	214,368

Figure: TABLE EXAMPLE IN SQL

Using the SQL SELECT Statement

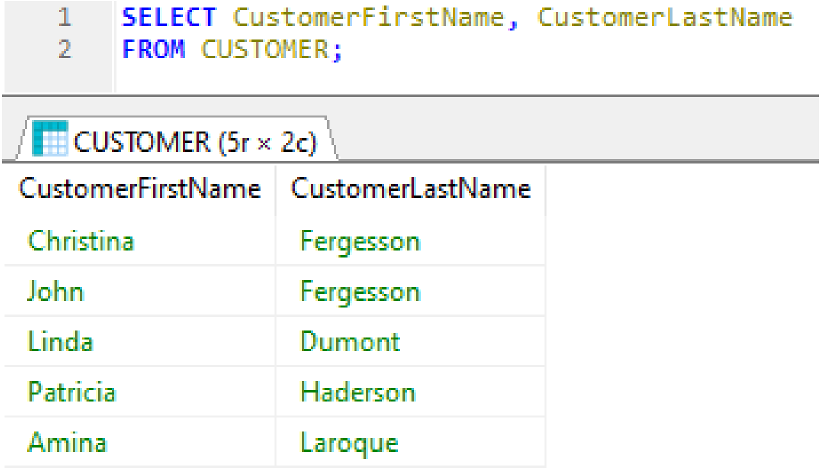
- The SELECT statement is used to query data from tables in a relational database management system (RDBMS).
- Basic syntax: `SELECT column1, column2, ... FROM tablename;`
- Example: `SELECT first_name, last_name FROM employees;`



```sql
SELECT *
FROM CUSTOMER;
```

CustomerId	CustomerFirstName	CustomerLastName	CustomerAddress	CustomerCity	CustomerCountry	CustomerPhoneNumber	AccountBalance
1	Christina	Fergesson	1234 First RD	London	UK	+44 1234 567890	125,000
2	John	Fergesson	1234 First RD	London	UK	+44 1234 556677	-536,952
3	Linda	Dumont	1234 BD de Lyon	Lille	France	+33 6 48 02 76 89	-32,588
4	Patricia	Haderson	9658 Green Street	Birmingham	UK	+44 5522 124698	1,528,884
5	Amina	Laroque	115 Av. de Valmy	Paris	France	+33 9 54 84 99 32	214,368

Figure: The select * statement

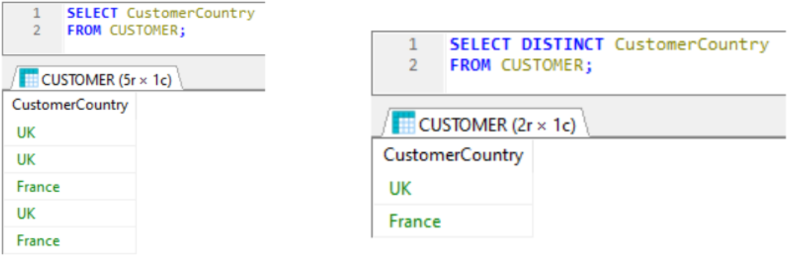


```sql
SELECT CustomerFirstName, CustomerLastName
FROM CUSTOMER;
```

CUSTOMER (5r × 2c)

CustomerFirstName	CustomerLastName
Christina	Fergesson
John	Fergesson
Linda	Dumont
Patricia	Haderson
Amina	Laroque

Figure: The select certain columns



```sql
SELECT CustomerCountry
FROM CUSTOMER;
```

CUSTOMER (5r × 1c)

```
CustomerCountry
UK
UK
France
UK
France
```

```sql
SELECT DISTINCT CustomerCountry
FROM CUSTOMER;
```

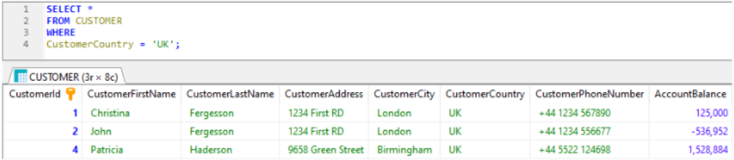
CUSTOMER (2r × 1c)

```
CustomerCountry
UK
France
```

Figure: Elimination of Duplicates

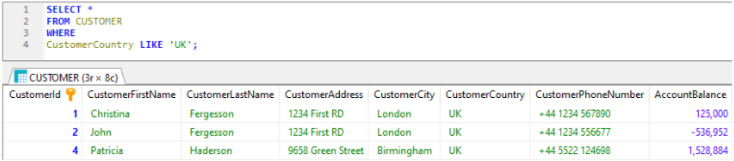
Filtering Results with the WHERE Clause

- The WHERE clause is used to filter query results based on specified conditions.
- Basic syntax: `SELECT column1, column2, ... FROM tablename WHERE condition;`
- Example: `SELECT * FROM employees WHERE department = 'HR';`



```sql
SELECT *
FROM CUSTOMER
WHERE
CustomerCountry = 'UK';
```

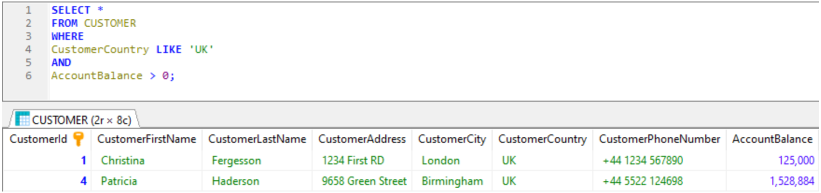
CustomerId	CustomerFirstName	CustomerLastName	CustomerAddress	CustomerCity	CustomerCountry	CustomerPhoneNumber	AccountBalance
1	Christina	Fergesson	1234 First RD	London	UK	+44 1234 567890	125,000
2	John	Fergesson	1234 First RD	London	UK	+44 1234 556677	-536,952
4	Patricia	Haderson	9658 Green Street	Birmingham	UK	+44 5522 124698	1,528,884



```sql
SELECT *
FROM CUSTOMER
WHERE
CustomerCountry LIKE 'UK';
```

CustomerId	CustomerFirstName	CustomerLastName	CustomerAddress	CustomerCity	CustomerCountry	CustomerPhoneNumber	AccountBalance
1	Christina	Fergesson	1234 First RD	London	UK	+44 1234 567890	125,000
2	John	Fergesson	1234 First RD	London	UK	+44 1234 556677	-536,952
4	Patricia	Haderson	9658 Green Street	Birmingham	UK	+44 5522 124698	1,528,884

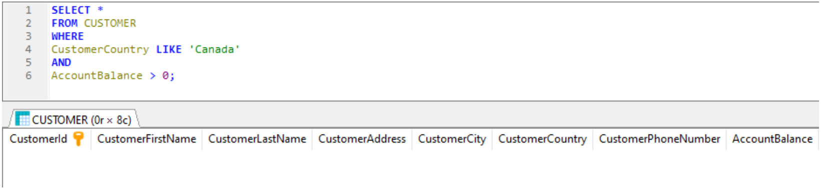
Figure: Filtering results



```sql
SELECT *
FROM CUSTOMER
WHERE
CustomerCountry LIKE 'UK'
AND
AccountBalance > 0;
```

CustomerId	CustomerFirstName	CustomerLastName	CustomerAddress	CustomerCity	CustomerCountry	CustomerPhoneNumber	AccountBalance
1	Christina	Fergesson	1234 First RD	London	UK	+44 1234 567890	125,000
4	Patricia	Haderson	9658 Green Street	Birmingham	UK	+44 5522 124698	1,528,884

Figure: Filtering results with logical operations

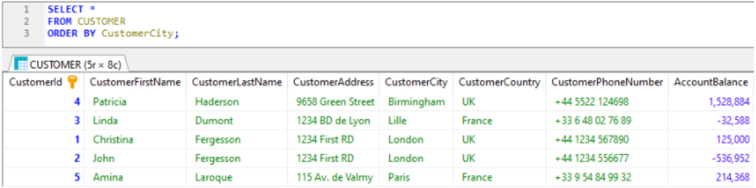


```sql
SELECT *
FROM CUSTOMER
WHERE
CustomerCountry LIKE 'Canada'
AND
AccountBalance > 0;
```

CUSTOMER (0r × 8c)

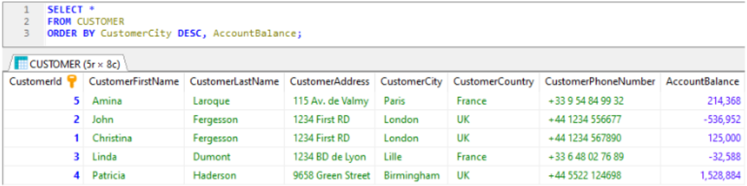
CustomerId	CustomerFirstName	CustomerLastName	CustomerAddress	CustomerCity	CustomerCountry	CustomerPhoneNumber	AccountBalance

Figure: Empty query result



```sql
SELECT *
FROM CUSTOMER
ORDER BY CustomerCity;
```

CustomerId	CustomerFirstName	CustomerLastName	CustomerAddress	CustomerCity	CustomerCountry	CustomerPhoneNumber	AccountBalance
4	Patricia	Haderson	9658 Green Street	Birmingham	UK	+44 5522 124698	1,528,884
3	Linda	Dumont	1234 BD de Lyon	Lille	France	+33 6 48 02 76 89	-32,588
1	Christina	Fergesson	1234 First RD	London	UK	+44 1234 567890	125,000
2	John	Fergesson	1234 First RD	London	UK	+44 1234 556677	-536,952
5	Amina	Laroque	115 Av. de Valmy	Paris	France	+33 9 54 84 99 32	214,368



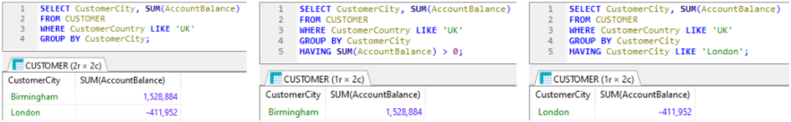
```sql
SELECT *
FROM CUSTOMER
ORDER BY CustomerCity DESC, AccountBalance;
```

CustomerId	CustomerFirstName	CustomerLastName	CustomerAddress	CustomerCity	CustomerCountry	CustomerPhoneNumber	AccountBalance
5	Amina	Laroque	115 Av. de Valmy	Paris	France	+33 9 54 84 99 32	214,368
2	John	Fergesson	1234 First RD	London	UK	+44 1234 556677	-536,952
1	Christina	Fergesson	1234 First RD	London	UK	+44 1234 567890	125,000
3	Linda	Dumont	1234 BD de Lyon	Lille	France	+33 6 48 02 76 89	-32,588
4	Patricia	Haderson	9658 Green Street	Birmingham	UK	+44 5522 124698	1,528,884

Figure: Sorting

Grouping Results with GROUP BY and HAVING

- The GROUP BY clause groups rows that have the same values in specified columns.
- The HAVING clause is used to filter the results of a GROUP BY based on a condition.
- Example: `SELECT department, COUNT(*) FROM employees GROUP BY department HAVING COUNT(*) > 5;`



```sql
SELECT CustomerCity, SUM(AccountBalance)
FROM CUSTOMER
WHERE CustomerCountry LIKE 'UK'
GROUP BY CustomerCity;
```

CUSTOMER (2r × 2c)

CustomerCity	SUM(AccountBalance)
Birmingham	1,528,884
London	-411,952

```sql
SELECT CustomerCity, SUM(AccountBalance)
FROM CUSTOMER
WHERE CustomerCountry LIKE 'UK'
GROUP BY CustomerCity
HAVING SUM(AccountBalance) > 0;
```

CUSTOMER (1r × 2c)

CustomerCity	SUM(AccountBalance)
Birmingham	1,528,884

```sql
SELECT CustomerCity, SUM(AccountBalance)
FROM CUSTOMER
WHERE CustomerCountry LIKE 'UK'
GROUP BY CustomerCity
HAVING CustomerCity LIKE 'London';
```

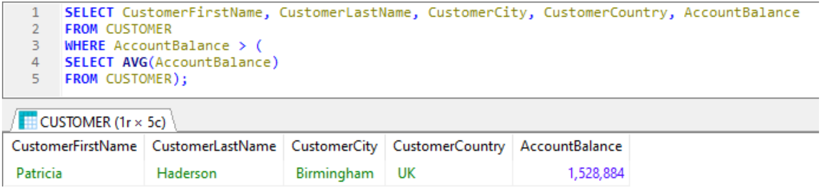
CUSTOMER (1r × 2c)

CustomerCity	SUM(AccountBalance)
London	-411,952

Figure: grouping

Understanding and Using Subqueries

- A subquery is a SQL query nested inside another query.
- Can be used in various parts of a SQL statement, such as SELECT, FROM, and WHERE clauses.
- Example: `SELECT employee_id, first_name FROM employees WHERE department_id = (SELECT department_id FROM departments WHERE department_name = ’HR’);`



```sql
SELECT CustomerFirstName, CustomerLastName, CustomerCity, CustomerCountry, AccountBalance
FROM CUSTOMER
WHERE AccountBalance > (
SELECT AVG(AccountBalance)
FROM CUSTOMER);
```

CustomerFirstName	CustomerLastName	CustomerCity	CustomerCountry	AccountBalance
Patricia	Haderson	Birmingham	UK	1,528,884

Figure: subqueries

EXPLAIN SIMPLY

1. What is the difference between data and information?
2. Why would you use queries instead of checking the entire contents of a table (2 reasons)?
3. What is the general form of an SQL SELECT statement?

1. Difference Between Data and Information

- **Data**: Raw facts and figures without context. For example, numbers, text, images, etc.
- **Information**: Processed data that has meaning and can lead to understanding.
- Data becomes information when it is processed and presented in a context that gives it value.

2. Using Queries Instead of Checking Entire Table

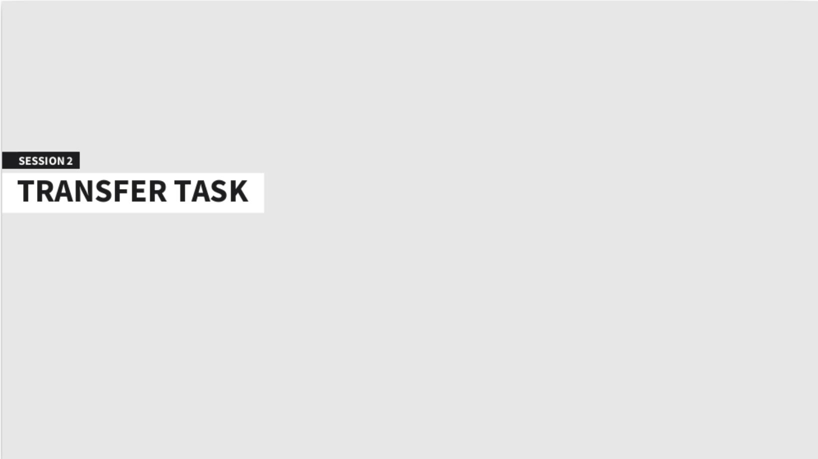
- **Efficiency**: Queries allow you to retrieve only the data you need, rather than loading the entire table, which can be resource-intensive.
- **Speed**: Retrieving specific data through a query can be much faster than manually searching through an entire table, especially for large datasets.

3. General Form of an SQL SELECT Statement

- The SELECT statement is used to retrieve data from one or more tables.
- **Syntax**: `SELECT column1, column2, ... FROM table_name WHERE condition;`
- Example: `SELECT first_name, last_name FROM employees WHERE department = ’HR’;`

Review study goals

- Use the SQL SELECT statement to query data in an Relational Database Management Syste (RDBMS).
- Filter query results using the WHERE clause.
- Group query results using the GROUP BY and HAVING clauses.
- Explain and use subqueries.



SESSION 2

TRANSFER TASK

Given the following ACCOUNT table:

AccountID	AccountType	FirstName	LastName	Balance
Acc001	Savings	Christina	Fergesson	125,000.00
Acc002	Chequing	Christina	Fergesson	14,526.00
Acc003	Business	John	Fergesson	523,621.00
Acc004	Business	Linda	Dumont	2,365,897.00
Acc005	Chequing	Patricia	Hadesson	-2,365.00
Acc006	Chequing	Amina	Laroque	11,425.00
Acc007	Chequing	Kelsey	Foster	-556.00
Acc008	Savings	Kelsey	Foster	10,265.00
Acc009	Business	Kelsey	Foster	2,589.00
Acc010	Savings	Mickey	Mohsen	-2,115.00
Acc011	Chequing	John	Doe	-55.00

Table: Accounts Information

Create queries that return:

1. Account type, account holder first name, account holder last name and account balance of all records
2. Account types without duplicates
3. First and last names of savings account holders
4. First and last names of negative balance accounts
5. First and last names of negative balance Savings accounts
6. The content of the table sorted by account type
7. The content of the table sorted by account type and reverse order of account balance
8. The maximum, minimum, and average of account balances
9. The sum and average of each account type
10. The sum of all accounts held by each customer (first and last name)
11. The sum of all negative balanced accounts held by each customer (first and last name)
12. Using a subquery, all data of negative balance accounts that are higher than the average negative account balances