



Module 3

Data Tier

Intro to SQL

Presented by
Domenico Mossali

IBM Client Innovation Center - Bari

Bari, 27/11/2023

IBM Client Innovation Center
Italy

Agenda



Intro to SQL

A Relational Database Overview

In short, SQL databases support SQL — a domain-specific language for querying and manipulating data in a relational database. The "relational" in a relational database refers to the "relational model" of data management devised by IBM researcher E.F. Codd in the early 1970s.

A database is a means of storing information in such a way that information can be retrieved from it.

In simplest terms, a **relational database** is one that presents information in tables with rows and columns.

A table is referred to as a relation in the sense that it is a collection of objects of the same type (rows). Data in a table can be related according to common keys or concepts, and the ability to retrieve related data from a table is the basis for the term relational database.

A Database Management System (DBMS) handles the way data is stored, maintained, and retrieved. In the case of a relational database, a Relational Database Management System (RDBMS) performs these tasks.

A Relational Database Overview #2

Integrity Rules

Relational tables follow certain integrity rules to ensure that the data they contain stay accurate and are always accessible. First, the rows in a relational table should all be distinct. If there are duplicate rows, there can be problems resolving which of two possible selections is the correct one. For most DBMSs, the user can specify that duplicate rows are not allowed, and if that is done, the DBMS will prevent the addition of any rows that duplicate an existing row.

A Relational Database Overview #3

Integrity Rules. Cont...

Another aspect of data integrity involves the concept of a null value. A database takes care of situations where data may not be available by using a null value to indicate that a value is missing. It does not equate to a blank or zero. *A blank is considered equal to another blank, a zero is equal to another zero, but two null values are not considered equal.*

When each row in a table is different, it is possible to use one or more columns to identify a particular row. This unique column or group of columns is called a **primary key**. Any column that is part of a primary key cannot be null; if it were, the primary key containing it would no longer be a complete identifier. This rule is referred to as entity integrity.

A Relational Database Overview #4

The Employees table illustrates some of these relational database concepts.

It has five columns and six rows, with each row representing a different employee.

Employees Table

Employee_Number	First_name	Last_Name	Date_of_Birth	Car_Number
10001	John	Washington	28-Aug-43	5
10083	Arvid	Sharma	24-Nov-54	null
10120	Jonas	Ginsberg	01-Jan-69	null
10005	Florence	Wojokowski	04-Jul-71	12
10099	Sean	Washington	21-Sep-66	null
10035	Elizabeth	Yamaguchi	24-Dec-59	null

A Relational Database Overview

#5

The primary key for this table would generally be the employee number because each one is guaranteed to be different. (A number is also more efficient than a string for making comparisons.) It would also be possible to use First_Name and Last_Name because the combination of the two also identifies just one row in our sample database. Using the last name alone would not work because there are two employees with the last name of "Washington." In this particular case the first names are all different, so one could conceivably use that column as a primary key, but it is best to avoid using a column where duplicates could occur. If Elizabeth Jones gets a job at this company and the primary key is First_Name, the RDBMS will not allow her name to be added (if it has been specified that no duplicates are permitted). Because there is already an Elizabeth in the table, adding a second one would make the primary key useless as a way of identifying just one row. Note that although using First_Name and Last_Name is a unique composite key for this example, it might not be unique in a larger database. Note also that the Employees table assumes that there can be only one car per employee.

Employees Table

Employee_Number	First_name	Last_Name	Date_of_Birth	Car_Number
10001	John	Washington	28-Aug-43	5
10083	Arvid	Sharma	24-Nov-54	null
10120	Jonas	Ginsberg	01-Jan-69	null
10005	Florence	Wojokowski	04-Jul-71	12
10099	Sean	Washington	21-Sep-66	null
10035	Elizabeth	Yamaguchi	24-Dec-59	null

From Oracle Java Documentation: <https://docs.oracle.com/javase/tutorial/jdbc/overview/database.html>

A Relational Database Overview #6

SELECT Statements

SQL is a language designed to be used with relational databases. There is a set of basic SQL commands that is considered standard and is used by all RDBMSs. For example, all RDBMSs use the SELECT statement.

A SELECT statement, also called a query, is used to get information from a table. It specifies one or more column headings, one or more tables from which to select, and some criteria for selection. The RDBMS returns rows of the column entries that satisfy the stated requirements. A SELECT statement such as the following will fetch the first and last names of employees who have company cars:

Employees Table

Employee_Number	First_name	Last_Name	Date_of_Birth	Car_Number
10001	John	Washington	28-Aug-43	5
10083	Arvid	Sharma	24-Nov-54	null
10120	Jonas	Ginsberg	01-Jan-69	null
10005	Florence	Wojokowski	04-Jul-71	12
10099	Sean	Washington	21-Sep-66	null
10035	Elizabeth	Yamaguchi	24-Dec-59	null

```
SELECT First_Name, Last_Name
FROM Employees
WHERE Car_Number IS NOT NULL
```


A Relational Database Overview #7

SELECT Statements. Cont...

The **result set** (the set of rows that satisfy the requirement of not having null in the Car_Number column) follows. The first name and last name are printed for each row that satisfies the requirement because the SELECT statement (the first line) specifies the columns First_Name and Last_Name. The FROM clause (the second line) gives the table from which the columns will be selected.

Employees Table

Employee_Number	First_name	Last_Name	Date_of_Birth	Car_Number
10001	John	Washington	28-Aug-43	5
10083	Arvid	Sharma	24-Nov-54	null
10120	Jonas	Ginsberg	01-Jan-69	null
10005	Florence	Wojokowski	04-Jul-71	12
10099	Sean	Washington	21-Sep-66	null
10035	Elizabeth	Yamaguchi	24-Dec-59	null

```
SELECT First_Name, Last_Name
FROM Employees
WHERE Car_Number IS NOT NULL
```

FIRST_NAME	LAST_NAME
John	Washington
Florence	Wojokowski

A Relational Database Overview #8

SELECT Statements. Cont...

The following code produces a result set that includes the whole table because it asks for all of the columns in the table Employees with no restrictions (no WHERE clause). Note that SELECT * means "SELECT all columns."

```
SELECT *  
FROM Employees
```

Employees Table

Employee_Number	First_name	Last_Name	Date_of_Birth	Car_Number
10001	John	Washington	28-Aug-43	5
10083	Arvid	Sharma	24-Nov-54	null
10120	Jonas	Ginsberg	01-Jan-69	null
10005	Florence	Wojokowski	04-Jul-71	12
10099	Sean	Washington	21-Sep-66	null
10035	Elizabeth	Yamaguchi	24-Dec-59	null

A Relational Database Overview

#9

WHERE Clauses

The WHERE clause in a SELECT statement provides the criteria for selecting values. For example, in the following code fragment, values will be selected only if they occur in a row in which the column Last_Name begins with the string 'Washington'.

```
SELECT First_Name, Last_Name
FROM Employees
WHERE Last_Name LIKE 'Washington%'
```

The keyword LIKE is used to compare strings, and it offers the feature that patterns containing wildcards can be used. For example, in the code fragment above, there is a percent sign (%) at the end of 'Washington', which signifies that any value containing the string 'Washington' plus zero or more additional characters will satisfy this selection criterion. So 'Washington' or 'Washingtonian' would be matches, but 'Washing' would not be.

Employees Table

Employee_Number	First_name	Last_Name	Date_of_Birth	Car_Number
10001	John	Washington	28-Aug-43	5
10083	Arvid	Sharma	24-Nov-54	null
10120	Jonas	Ginsberg	01-Jan-69	null
10005	Florence	Wojokowski	04-Jul-71	12
10099	Sean	Washington	21-Sep-66	null
10035	Elizabeth	Yamaguchi	24-Dec-59	null

A Relational Database Overview

#10

The other wildcard used in LIKE clauses is an underbar (_), which stands for any one character. For example,

```
WHERE Last_Name LIKE 'Ba_man'
```

would match 'Barman', 'Badman', 'Balman', 'Bagman', 'Bamman', and so on.

The code fragment below has a WHERE clause that uses the equal sign (=) to compare numbers. It selects the first and last name of the employee who is assigned car 12.

```
SELECT First_Name, Last_Name
FROM Employees
WHERE Car_Number = 12
```

Employees Table

Employee_Number	First_name	Last_Name	Date_of_Birth	Car_Number
10001	John	Washington	28-Aug-43	5
10083	Arvid	Sharma	24-Nov-54	null
10120	Jonas	Ginsberg	01-Jan-69	null
10005	Florence	Wojokowski	04-Jul-71	12
10099	Sean	Washington	21-Sep-66	null
10035	Elizabeth	Yamaguchi	24-Dec-59	null

A Relational Database Overview

#11

The next code fragment selects the first and last names of employees whose employee number is greater than 10005:

```
SELECT First_Name, Last_Name
FROM Employees
WHERE Employee_Number > 10005
```

Employees Table

Employee_Number	First_name	Last_Name	Date_of_Birth	Car_Number
10001	John	Washington	28-Aug-43	5
10083	Arvid	Sharma	24-Nov-54	null
10120	Jonas	Ginsberg	01-Jan-69	null
10005	Florence	Wojokowski	04-Jul-71	12
10099	Sean	Washington	21-Sep-66	null
10035	Elizabeth	Yamaguchi	24-Dec-59	null

WHERE clauses can get rather elaborate, with multiple conditions and, in some DBMSs, nested conditions. This overview will not cover complicated WHERE clauses, but the following code fragment has a WHERE clause with two conditions; this query selects the first and last names of employees whose employee number is less than 10100 and who do not have a company car.

```
SELECT First_Name, Last_Name
FROM Employees
WHERE Employee_Number < 10100 and Car_Number IS NULL
```

A Relational Database Overview

Joins

A distinguishing feature of relational databases is that it is possible to get data from more than one table in what is called a **join**. Suppose that after retrieving the names of employees who have company cars, one wanted to find out who has which car, including the license plate number, mileage, and year of car. This information is stored in another table, Cars:

There must be one column that appears in both tables in order to relate them to each other. This column, which must be the primary key in one table, is called the foreign key in the other table. In this case, the column that appears in two tables is Car_Number, which is the primary key for the table Cars and the foreign key in the table Employees.

From Oracle Java Documentation: <https://docs.oracle.com/javase/tutorial/jdbc/overview/database.html>

November © 2023 IBM Client Innovation Center – Java Academy 2023 – Module 3 – Data Tier

#12

Employees Table

Employee_Number	First_name	Last_Name	Date_of_Birth	Car_Number
10001	John	Washington	28-Aug-43	5
10083	Arvid	Sharma	24-Nov-54	null
10120	Jonas	Ginsberg	01-Jan-69	null
10005	Florence	Wojokowski	04-Jul-71	12
10099	Sean	Washington	21-Sep-66	null
10035	Elizabeth	Yamaguchi	24-Dec-59	null

Cars Table

Car_Number	License_Plate	Mileage	Year
5	ABC123	5000	1996
12	DEF123	7500	1999

A Relational Database Overview

Joins. Cont...

If the 1996 car with license plate number ABC123 were wrecked and deleted from the Cars table, then Car_Number 5 would also have to be removed from the Employees table in order to maintain what is called referential integrity. Otherwise, the **foreign key** column (Car_Number) in the Employees table would contain an entry that did not refer to anything in the Cars table. A foreign key must either be null or equal to an existing primary key value of the table to which it refers. This is different from a primary key, which may not be null. There are several null values in the Car_Number column in the table Employees because it is possible for an employee not to have a company car.

#13

Employees Table

Employee_Number	First_name	Last_Name	Date_of_Birth	Car_Number
10001	John	Washington	28-Aug-43	5
10083	Arvid	Sharma	24-Nov-54	null
10120	Jonas	Ginsberg	01-Jan-69	null
10005	Florence	Wojokowski	04-Jul-71	12
10099	Sean	Washington	21-Sep-66	null
10035	Elizabeth	Yamaguchi	24-Dec-59	null

Cars Table

Car_Number	License_Plate	Mileage	Year
5	ABC123	5000	1996
12	DEF123	7500	1999

A Relational Database Overview

#14

Joins. Cont...

The following code asks for the first and last names of employees who have company cars and for the license plate number, mileage, and year of those cars. Note that the FROM clause lists both the Employees and Cars tables because the requested data is contained in both tables. Using the table name and a dot (.) before the column name indicates which table contains the column.

```
SELECT Employees.First_Name, Employees.Last_Name,  
       Cars.License_Plate, Cars.Mileage, Cars.Year  
FROM Employees, Cars  
WHERE Employees.Car_Number = Cars.Car_Number
```

Employees Table

Employee_Number	First_name	Last_Name	Date_of_Birth	Car_Number
10001	John	Washington	28-Aug-43	5
10083	Arvid	Sharma	24-Nov-54	null
10120	Jonas	Ginsberg	01-Jan-69	null
10005	Florence	Wojokowski	04-Jul-71	12
10099	Sean	Washington	21-Sep-66	null
10035	Elizabeth	Yamaguchi	24-Dec-59	null

Cars Table

Car_Number	License_Plate	Mileage	Year
5	ABC123	5000	1996
12	DEF123	7500	1999

A Relational Database Overview

Joins. Cont...

```
SELECT Employees.First_Name, Employees.Last_Name,
       Cars.License_Plate, Cars.Mileage, Cars.Year
FROM Employees, Cars
WHERE Employees.Car_Number = Cars.Car_Number
```

This returns a result set that will look similar to the following:

FIRST_NAME	LAST_NAME	LICENSE_PLATE	MILEAGE	YEAR
John	Washington	ABC123	5000	1996
Florence	Wojokowski	DEF123	7500	1999

#15

Employees Table

Employee_Number	First_name	Last_Name	Date_of_Birth	Car_Number
10001	John	Washington	28-Aug-43	5
10083	Arvid	Sharma	24-Nov-54	null
10120	Jonas	Ginsberg	01-Jan-69	null
10005	Florence	Wojokowski	04-Jul-71	12
10099	Sean	Washington	21-Sep-66	null
10035	Elizabeth	Yamaguchi	24-Dec-59	null

Cars Table

Car_Number	License_Plate	Mileage	Year
5	ABC123	5000	1996
12	DEF123	7500	1999

SQL Suggested References

<https://www.w3schools.com/sql/>