Databse - Introduction

- A database is an organized collection of data.
- A database management system (DBMS) provides mechanisms for storing, organizing, retrieving and modifying data for many users.
- Database management systems allow for the access and storage of data without concern for the internal representation of data.

Popular Relational Database Management Systems

- Some popular proprietary relational database management systems (RDBMSs) are Microsoft SQL Server, Oracle, Sybase and IBM DB2.
- PostgreSQL, MariaDB and MySQL are popular open-source DBMSs that can be downloaded and used freely by anyone.
- JDK 8 comes with a pure-Java RDBMS called Java DB—the Oracle-branded version of Apache DerbyTM.
- As of JDK 9, Oracle no longer bundles Java DB with the JDK.

JDBC

- Java programs interact with databases using the Java Database Connectivity (JDBCTM) API.
- A JDBC driver enables Java applications to connect to a database in a particular DBMS and allows you to manipulate that database using the JDBC API.

Relational Databases

- A relational database is a logical representation of data that allows the data to be accessed without consideration of its physical structure.
- A relational database stores data in tables.

A Book Database

- We introduce relational databases in the context of a books database.
- We use this database to introduce various database concepts, including how to use SQL to obtain information from the database and to manipulate the data.
- The database consists of three tables: Authors, AuthorISBN and Titles.

Authors Table

- The Authors table consists of three columns that maintain each author’s unique ID number, first name and last name.
- AuthorID : This integer column is defined as auto_increment. This is the primary key.
- FirstName : Author’s first name (a string)
- LastName : Author’s last name (a string)

Titles Table

- The Titles table consists of four columns that maintain information about each book in thedatabase, including its ISBN, title, edition number and copyright year.
- ISBN : ISBN of the book (a string). Primary key. (“International Standard Book Number”)
- Title : Title of the book (a string).
- EditionNumber : Edition number of the book (an integer).
- Copyright : Copyright year of the book (a string).

AuthorISBN Table

- The AuthorISBN table consists of two columns that maintain ISBNs for each book and theircorresponding authors' ID numbers.
- This table associates authors with their books.
- The AuthorID column is a foreign key—a column in this table that matches the primary-key column in another table (that is, AuthorID in the Authors table).

AuthorISBN Table (contd.)

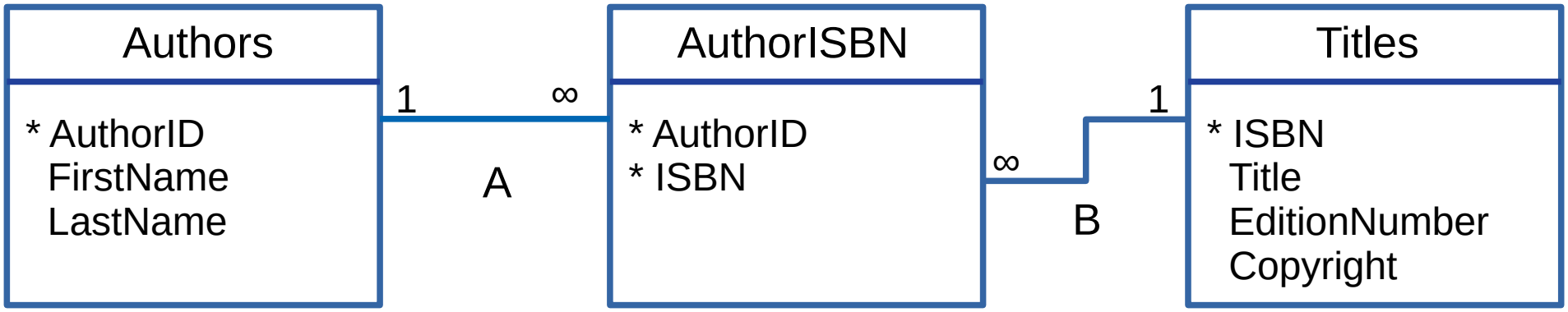
- The ISBN column is also a foreign key—it matches the primary-key column (that is, ISBN) in the Titles table.
- A database might consist of many tables.
- A goal when designing a database is to minimize the amount of duplicated data among the database's tables.
- Foreign keys, which are specified when a database table is created in the database, link the data in multiple tables.
- Together the AuthorID and ISBN columns in this table form a composite primary key.

AuthorISBN Table (contd.)

- Every row in this table uniquely matches one author to one book’s ISBN.
- Every foreign-key value must appear as another table’s primary-key value so the DBMS can ensure that the foreignkey value is valid—this is known as the Rule of Referential Integrity.
- For example, the DBMS ensures that the AuthorID value for a particular row of the AuthorISBN table is valid by checking that there is a row in the Authors table with that AuthorID as the primary key.

- Foreign keys also allow related data in multiple tables to be selected from those tables—this is known as joining the data.
- There is a one-to-many relationship between a primary key and a corresponding foreign key (for example, one author can write many books and one book can be written by many authors). This means that a foreign key can appear many times in its own table but only once (as the primary key) in another table.
- For example, the ISBN 0132151006 can appear in several rows of AuthorISBN (because this book has several authors) but only once in Titles, where ISBN is the primary key.

Entity-Relationship (ER) Diagram



ER diagram for the books database

- This diagram shows the database tables and the relationships among them.
- Primary keys are identified by star (*).

- A table’s primary key uniquely identifies each row in the table.
- Every row must have a primary-key value, and that value must be unique in the table. This is known as the Rule of Entity Integrity.
- For the AuthorISBN table, the primary key is the combination of both columns—this is known as a composite primary key.
- The line A connecting the tables Authors and AuthorISBN with values 1 and ∞ at each end indicates a one-to-many relationship.

- That is, for each author in the Authors table, there can be an arbitrary number of ISBNs for books written by that author in the AuthorISBN table (that is, an author can write any number of books).
- The line B between the Titles and AuthorISBN tables illustrates a one-to-many relationship — one book can be written by many authors.
- The relationships in Figure illustrate that the sole purpose of the Author-ISBN table is to provide a many-to-many relationship between the Authors and Titles tables — an author can write many books, and a book can have many authors.

SQL (Structured Query Language)

SQL Keyword	Description
Select	Retrieves data from one or more tables.
from	Tables involved in the query. Required in every SELECT.
where	Criteria for selection that determine the rows to be retrieved, deleted or updated. Optional in a SQL statement.

• group by	• Criteria for grouping rows. Optional in a SELECT query.
• order by	• Criteria for ordering rows. Optional in a SELECT query.
• inner join	• Merge rows from multiple tables.
• insert	• Merge rows from multiple tables.
• update	• Update rows in a specified table.
• delete	• Delete rows from a specified table.

Basic SELECT Query

- Retrieves rows from one or more tables in a database.

```
select * from tableName
```

- The asterisk (*) wildcard character indicates that all columns from the tableName table should be retrieved.
- For example, to retrieve all the data in the Authors table, use

```
select * from Authors;
```

Basic SELECT Query (contd.)

- To retrieve only specific columns, replace the * with a comma separated list of column names.
- For example, to retrieve only the columns AuthorID and LastName for all rows in the Authors table, use the query:

```
select AuthorID, LastName, FirstName
   from Authors;
```

where clause

- SQL uses the optional WHERE clause in a query to specify the selection criteria for the query. The basic form of a query with selection criteria is:

  select colName1, colName2, … from tableName
  where criteria

```
select Title, EditionNumber, Copyright
  from Titles
  where Copyright > ‘2013’;
```

- Strings in SQL are delimited by single (') rather than double (") quotes.

Pattern Matching: Zero or More Characters

- The WHERE clause criteria can contain the operators <, >, <=, >=, =, <> (not equal) and LIKE.
- Operator LIKE is used for pattern matching with wildcard characters percent (%) and underscore (_).
- Pattern matching allows SQL to search for strings that match a given pattern.
- A pattern that contains a percent character (%) searches for strings that have zero or more characters at the percent character’s position in the pattern.

- For example, the next query locates the rows of all the authors whose last name starts with the letter D:

```
select AuthorID, FirstName, LastName

from Authors

where LastName like 'D%';
```

- Pattern maching: Any character
- An underscore (_) in the pattern string indicates a single wildcard character at that position in the pattern.

- For example, the following query locates the rows of all the authors whose last names start with any character (specified by _), followed by the letter o, followed by any number of additionalcharacters (specified by %):

```
select AuthorID, FirstName, LastName

from Authors

where LastName like '_o%';
```

order by clause

- The rows in the result of a query can be sorted into ascending or descending order by using the optional ORDER BY clause. The basic form of a query with an ORDER BY clause is:

  select colName1, colName2, ... from tableName
  order by colName asc

  select colName1, colName2, ... from tableName
  order by colName desc

- For example, to obtain the list of authors in ascending order by last name, use the query:

```
select AuthorID, FirstName, LastName

from Authors

order by LastName asc;
```

- The default sorting order is ascending, so ASC is optional.

Sorting By Multiple Columns

- Basicfor:

  order by colName1 sortingOrder, colName2 sortingOrder, ...

- Where sortingOrder is either asc or desc. The sorting order need not be identical for each column.

```
Select AuthorID, FirstName, LastName
   from Authors
   order by LastName, FirstName;
```

- Sorts all the rows in ascending order by last name, then by first name.

Combining the WHERE and ORDER BY Clauses

```
select ISBN, Title, EditionNumber, Copyright
from Titles
where Title like '%How to Program'
order by Title asc;
```

- The above query returns the ISBN, Title, EditionNumber and Copyright of each book in the Titles table that has a Title ending with "How to Program" and sorts them in ascending order by Title.

Merging Data from Multiple Tables: INNER JOIN

- Database designers often split related data into separate tables to ensure that a database does not store data redundantly.
- Often, it’s necessary to merge data from multiple tables into a single result.
- Referred to as joining thetables, this is specified by an INNER JOIN operator, which merges rows from two tables by matching values in columns that are common to the tables.
- The basic form of an INNER JOIN is:

Merging Data from Multiple Tables: INNER JOIN

- select columnName1, columnName2, ..
  from table1
  inner join table2
  on table1.columnName = table2.columnName
- The ON clause of the INNER JOIN specifies the columns from each table that are compared to determine which rows are merged—one is a primary key and the other is a foreign key in the tables being joined.

- For example, the following query produces a list of authors accompanied by the ISBNs for books written by each author:

```
select FirstName, LastName, ISBN
from Authors
inner jon AuthorISBN
on Authors.AuthorID = AuthorISBN.AuthorID
order by LastName, FirstName;
```

- The query merges the FirstName and LastName columns from table Authors with the ISBN column from table AuthorISBN, sorting the results in ascending order by LastName and FirstName.
- Note the use of the syntax tableName.columnName in the ON clause. This syntax, called a qualified name, specifies the columns from each table that should be compared to join the tables.

INSERT Statement

- The INSERT statement inserts a row into a table. The basic form of this statement is

  ```
  insert into tableName (columnName1,
      columnName2, ..., columnNameN)
  values (value1, value2, ..., valueN)
  ```

- The values specified here must match the columns specified after the table name in both order and type.
- Always explicitly list the columns when inserting rows.

- If the table’s column order changes or a new column is added, using only VALUES may cause an error.
- The INSERT statement

```
insert into Authors (FirstName, LastName)
   values ('Sue', 'Red');
```

- inserts a row into the Authors table.
- We do not specify an AuthorID in this example because AuthorID is an autoincremented column in the Authors table.

Common Programming Errors

- Note: SQL delimits strings with single quotes ('). A string containing a single quote (e.g., Sun’s rays) must have two single quotes in the position where the single quote appears (e.g., `'Sun''s rays'`). The first acts as an escape character for the second. Not escaping single-quote characters in a string that’s part of a SQL statement is a SQL syntax error.
- It’s normally an error to specify a value for an autoincrement column.

UPDATE Statement

- An UPDATE statement modifies data in a table. Its basic form is:

```
update tableName
    set columnName1 = value1, columnName2 =
        value2, ..., columnNameN = valueN
where criteria
```

- where tableName is the table to update.

UPDATE statement (cont.)

```
update Authors
 set LastName = 'Black'
 where LastName = 'Red' and FirstName = 'Sue';
```

- updates a row in the Authors table.
- If we know the AuthorID, then *where* clause can be simlified as:

```
  where AuthorID = 6
```

Delete Statement

- A SQL DELETE statement removes rows from a table. Its basic form is:
- DELETE FROM tableName WHERE criteria
- where tableName is the table from which to delete.
- The optional WHERE clause specifies the criteria used to determine which rows to delete.
- If this clause is omitted, all the table’s rows are deleted.

Delete Statement (contd.)

- The DELETE statement:

```
delete from Authors
 where LastName = 'Black' and FirstName =
'Sue';
```

- deletes the row for Sue Black in the Authors table.
- If we know the AuthorID in advance of the DELETE operation, the WHERE clause can be simplified as follows:

```
where AuthorID = 6
```

```
mysql -u root -p

create user 'student'@'localhost'
  identified by 'Student10*';

create database stest;

grant all privileges on stest.* to
  'student'@'localhost';

flush privileges;

mysql -u student -p

select now();

select 200 + 100;
```

```
show databases;
select stest;
source books.sql;
show tables;
describe Authors;
select * from Authors;
select ISNB, Title from Titles;
set password = 'NewPassword1*';
```