COMP3278 Introduction to Database Management Systems

In this tutorial we try to practice the implementation of a database system. We are going to implement a book catalog system. The following is the description of the application:

* We store **books** in the database.
  + Each book has a ***bookID*** (unique identifier), ***name***, ***description***, ***price*** and ***picture***.
* We store **authors** in the database.
  + Each author has an ***authorID*** (unique identifier) and ***name*** .
* Authors can write many books, a book can be written by many authors.
* Each book can belong to many **categories** (e.g., Computer Science, Algorithm, Story …etc)
* Assuming partial participation in all cases in this system.

**Section 1. Information modeling (Chapter 2A).**

We first draw an ER diagram that model the data in the application.

图示

AI 生成的内容可能不正确。

**Section 2. Translate to database tables and build it. (Chapter 2B and Tutorial 2).**

* Let’s create a new table under your book database
* Let’s name the database table t3\_books, we use prefix “t3\_” to mark the table name in this tutorial in order not to mix up the tables we created in previous tutorials.
* Click the Go button to add one more columns in the table, you can modify the table structure anytime.
* Add the table attribute according to the following table.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Table name** | **Storage engine** | **Column Name** | **Type** | **Length/Values** | **A\_I** | **Primary key** |
| t3\_books | InnoDB | bookID | INT | 12 | Yes | Yes |
| name | VARCHAR | 100 | / | / |
| description | TEXT |  | / | / |
| price | DOUBLE |  | / | / |
| picture | TEXT |  | / | / |

* **What is A\_I?** For the **bookID** column, check also the A\_I option, which stands for “auto increment”. This option allows you to insert a new tuple in the table without specifying **bookID**, and it will automatically assign a unique **bookID** for the new tuple.
* **What is InnoDB?** This is the storage engine that handles how data are physically stored. Only **InnoDB** storage engine supports the checking of referential constraints in MySQL. Please select the storage engine of this table to be InnoDB.
* Please create the following 4 more tables in the database.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Table name** | **Storage engine** | **Attribute Name** | **Type** | **Length/Values** | **A\_I** | **Primary key** |
| t3\_authors | InnoDB | authorID | INT | 12 | Yes | Yes |
| name | VARCHAR | 100 | / | / |
| t3\_writes | InnoDB | bookID | INT | 12 | / | Yes |
| authorID | INT | 12 | / | Yes |
| t3\_categories | InnoDB | categoryID | INT | 12 | Yes | Yes |
| name | VARCHAR | 100 | / | / |
| t3\_belongs | InnoDB | bookID | INT | 12 | / | Yes |
| categoryID | INT | 12 | / | Yes |

* Creating referential constraints – We have the following referential constraints:
  + **t3\_write** table’s **bookID** references **t3\_book** table’s **bookID**
  + **t3\_write** table’s **authorID** references **t3\_author** table’s **authorID**
  + **t3\_belongs** table’s **bookID** references **t3\_book** table’s **bookID**
  + **t3\_belongs** table’s **categoryID** references **t3\_categories** table’s **categoryID**
* Execute the following SQLs to create the referential constraint in MySQL.

|  |
| --- |
| **ALTER TABLE** t3\_writes **ADD FOREIGN KEY** (bookID) **REFERENCES** t3\_books (bookID);  **ALTER TABLE** t3\_writes **ADD FOREIGN KEY** (authorID) **REFERENCES** t3\_authors (authorID);  **ALTER TABLE** t3\_belongs **ADD FOREIGN KEY** (bookID) **REFERENCES** t3\_books (bookID);  **ALTER TABLE** t3\_belongs **ADD FOREIGN KEY** (categoryID) **REFERENCES** t3\_categories (categoryID);  **ALTER TABLE** t3\_belongs **ADD FOREIGN KEY** (categoryID) **REFERENCES** t3\_categorie |

* You can do it by selecting your database on the left panel > Click the SQL tab> Copy and paste the SQL in the above table to the SQL area in the interface > Click Go to execute the SQLs.

图形用户界面, 应用程序

AI 生成的内容可能不正确。

* To check if the referential constraints are working, let’s insert a record “**1,1**” in the **t3\_writes** table.
  + You should see the following output, which means that inserting **bookID** of 1 and **authorID** of 1 in **t3\_writes** table violates referential constraints.
  + We don’t have book with **bookID** equals 1 in the **t3\_books** table
  + We don’t have authors with **authorID** equals to 1 in the **t3\_author** table.

图形用户界面, 文本, 应用程序

AI 生成的内容可能不正确。

* Import tuples into the database tables (import comp3278.sql)
* You can do it by selecting your database in the left panel > Click the Import tab > Browse your computer > Select the **comp3278.sql** file > Click Go to import.

图形用户界面, 文本, 应用程序, 电子邮件

AI 生成的内容可能不正确。

* Now you can browse the tables and see there are some sample tuples inserted.

**SQL Practice**

**Exercise 1: Basic Queries**

**Question**: Retrieve all book titles and their prices.

SELECT name, price FROM t3\_books;

**Exercise 2: Join Operations**

**Question**: List all books along with their authors.

SELECT t3\_books.name, t3\_authors.name FROM t3\_books, t3\_authors, t3\_writes WHERE t3\_books.bookID = t3\_writes.bookID AND t3\_writes.authorID = t3\_authors.authorID;

**Exercise 3: Aggregation Functions**

**Question**: Count how many books each author has written.

select A.name, count(\*) from t3\_authors A, t3\_writes W

where A.authorID = W.authorID

group by A.name;

**Exercise 4: Complex Filtering**

**Question**: Find all books that belong to both "Computer Science" and "Database" categories.

SELECT

B.bookID,

B.name,

B.description,

B.price,

B.picture,

GROUP\_CONCAT(C.name SEPARATOR ', ') AS categories

FROM t3\_books B, t3\_categories C

WHERE C.name IN ("Computer Science", "Database")

GROUP BY B.bookID, B.name, B.description, B.price, B.picture;

**Exercise 5: Subqueries**

**Question**: Find books with prices higher than the average book price.

select \* from t3\_books B

where B.price > (select avg(B.price) from t3\_books B);

nested query is needed

**Exercise 6: Multi-table Joins**

**Question**: List all books with their categories and authors.

select distinct B.name as 'book name',

GROUP\_CONCAT(distinct C.name SEPARATOR ', ') as 'categories',

GROUP\_CONCAT(distinct A.name SEPARATOR ', ') as 'authors'

from t3\_books B, t3\_categories C, t3\_authors A, t3\_writes W, t3\_belongs Be

where B.bookID = W.bookID

and W.authorID = A.authorID

and Be.bookID = B.bookID

and C.categoryID = Be.categoryID

group by B.name;