

DSAA 5012: Advanced Database Management for Data Science

Lecture 8 Exercises SQL Queries

Book(bid, title, subject, quantityInStock, price, aid)

Author(aid, firstName, lastName)

Customer(cid, firstName, lastName)

BookOrder(oid, cid, orderYear)

OrderDetails(oid, bid, quantity)

Exercise 1: Given the foreign keys of the Book Store relations and assuming the referential integrity constraints are included in the SQL create statements, what should be the create order?

Relation	Create Order
Author	_____
Customer	_____
Book	_____
BookOrder	_____
OrderDetails	_____

Exercise 2: For all authors who wrote books on at least two subjects, increase the price of all their books by 5%.

Exercise 3: Find the last name and first name of all authors who wrote books on both the subjects of Art and Business.

Name: (1) _____ / _____ Student#: (1) _____ Date: _____
Last/Family (PRINT) Given/First (PRINT)

Name: (2) _____ / _____ Student#: (2) _____
Last/Family (PRINT) Given/First (PRINT)

NOTE: You are highly encouraged to do this exercise with a partner.

DSAA 5012: Advanced Database Management for Data Science

Lecture 8 Exercises SQL Queries

NOTE: Use only SQL constructs discussed in the lectures to answer these queries.

Book(bid, title, subject, quantityInStock, price, *aid*) Author(aid, firstName, lastName)

Customer(cid, firstName, lastName) BookOrder(oid, *cid*, orderYear) OrderDetails(oid, bid, quantity)

Exercise 4: Find the last name and first name of all authors who wrote books on exactly ten different subjects. **Do not use subqueries; do not create any derived relations.**

Exercise 5: For each customer who made more than 10 orders in 2019, find the customer id, last name and the number of orders in 2019. **Do not use subqueries; do not create any derived relations.**

Exercise 6: Find the customer id, last name and total quantity ordered for those customers who ordered the largest total quantity of books.

Upload this completed exercise worksheet to Canvas by 11 p.m. Feb 26th.

Name: (1) _____ / _____ Student#: (1) _____ Date: _____
Last/Family (PRINT) Given/First (PRINT)

Name: (2) _____ / _____ Student#: (2) _____
Last/Family (PRINT) Given/First (PRINT)

NOTE: You are highly encouraged to do this exercise with a partner.

Branch(branchName, district, assets) Account(accountNo, balance, *branchName*) Borrower(clientId, loanNo)
Client(clientId, name, address, district) Loan(loanNo, amount, *branchName*) Depositor(clientId, accountNo)

Exercise 7: The following PL/SQL procedure is used to calculate the interest payable to an account and to update the account balance with the interest payable according to the following schedule.

- 0% if balance < \$10,000
- 2% if \$10,000 ≤ balance < \$100,000
- 4% if balance ≥ \$100,000.

Additionally, if the account balance is greater than or equal to \$100,000 and the client holding the account has a loan, then an additional 1% interest is given.

Complete the accountCursor and borrowerCursor definitions so that the PL/SQL procedure executes correctly.

create or replace procedure CalculateInterest **as**

```
currentAccountNo Account.accountNo%type;  
interestPayable Account.balance%type;  
percentInterest number;
```

-- The cursor for the Account table

cursor accountCursor **is** _____

-- The cursor for the join of the Borrower and Depositor tables for the current account
cursor borrowerCursor **is** _____

Complete these cursor definitions.

begin

for accountRecord **in** accountCursor **loop**

```
currentAccountNo := accountRecord.accountNo;
```

```
-- Determine the percent interest to pay
```

```
percentInterest := 0;
```

```
if (accountRecord.balance >= 10000 and accountRecord.balance < 100000) then
```

```
percentInterest := 0.02;
```

```
elsif (accountRecord.balance >= 100000) then percentInterest := 0.04;
```

```
-- Give an additional 1% interest if the client has a loan
```

```
for borrowerRecord in borrowerCursor loop
```

```
if (borrowerRecord.numLoans <> 0) then
```

```
percentInterest := percentInterest + 0.01;
```

```
end if;
```

```
end loop;
```

```
end if;
```

```
-- Calculate the interest payable
```

```
interestPayable := accountRecord.balance * percentInterest;
```

```
-- Update the client's account balance
```

```
update Account set balance = balance + interestPayable
```

```
where accountNo=currentAccountNo;
```

```
end loop;
```

```
end CalculateInterest;
```

Upload this completed exercise worksheet to Canvas by 11 p.m. Feb 26th.