Institut für Informatik - Lehrstuhl Datenbanken

Einführung in Datenbanken

Wintersemester 22/23 - Prof. Dr. Stefan Brass, PD Dr. Alexander Hinneburg

Übung 10: SQL-Anfragen Aggregation, Relationale Algebra
Abgabe bis Mo. 16.1.2023, 18:00 Uhr,
Übungsplattform: Studip → Einführung in Datenbanken → Übungsplattform

Aufgabe 10.1: 6 Punkte

Die kleine Beispieldatenbank beschreibt mit ihren Tabellen Studierende, Kurse, Einschreibungen von Studierenden für Kurse, Bücher und Buchempfehlungen für Kurse. Die Typen der Attribute sind hinter den Namen dargestellt. Diese Datenbank gibt es nur für Übungen für Relationale Algebra mit dem Werkzeug RelaX. Es gibt keine Postgres-Version dieser Datenbank.

- STUDENT(SSN:number, Name:string, Major:string, Bdate:string)
- COURSE(CourseId:number, Cname:string, Dept:string)
- ENROLL(SSN:number→ STUDENT, CourseId:number→ COURSE, Quarter:string), Grade°:number)
- BOOK_RECOMMENDATION(CourseId:number→ COURSE, Quarter:string, Book_ISBN:string→BOOK)
- BOOK(Book_ISBN:string, Book_Title:string, Publisher:string, Author:string)

Die Datenbank ist mit RelaX unter DBS1 Student Course Enroll zu erreichen:
`http://dbis-uibk.github.io/relax/calc/gist/dd9b9e4a5bd3b9a5265104e4c8f171c6/student_course_enroll/0`

Geben Sie für die folgenden Anfragen entsprechende Statements in relationaler Algebra an. Für jede Anfrage soll nur ein Statement angegeben werden, das jedoch Zwischenergebnisse enthalten kann. Falls die Tutoren eine Anfrage nicht schnell verstehen können, kann es Punktabzug geben. Sie können das durch gute Formatierung und hilfreiche Kommentare (wo nötig) vermeiden. Beachten Sie aber, dass die Tutoren relationale Algebra natürlich sehr gut kennen. Vermeiden Sie triviale Kommentare. Statements mit Syntax-Fehlern in RelaX werden mit Null Punkten bewertet. Geben Sie jede Teilaufgabe als separate Text-Datei mit der Endung .txt ab (kein Word). Unter Windows können Sie Text-Dateien mit Notepad editieren. Das Programm Notepad++ (`http://notepad-plus-plus.org`) bietet mehr Funktionen.

a) Geben sie alle Bücher mit ISBN und Titel aus, die Daniel mit der SSN 931245 im Wintersemester 2020 (WiSe 2020) in irgendwelchen Kursen empfohlen wurden?

Geben Sie das Buch mit ISBN und Titel sowie den Kurs mit CourseID und Namen aus in dem das Buch empfohlen wurde.

b) Welche Studierenden bekamen das Buch 'Algorithmen und Datenstrukturen' mit der ISBN '978-3495819459' in einem Kurs empfohlen? Geben Sie die SSN und den Namen der Studierenden aus.

c) Welche Studierenden haben den Kurs 'Web-Programmierung' (noch) nicht besucht? Geben Sie die SSN und den Namen der Studierenden aus.

Aufgabe 10.2: 6 Punkte

Das folgende relationale Schema beschreibt eine Film-Verleih-Kette. Das Datenbank-Schema wurde 2006 von MySQL zu Demonstrationszwecken entwickelt. Eine Beschreibung finden Sie unter `http://dev.mysql.com/doc/sakila/en/index.html`. Loggen Sie sich in die PostgreSQL-Datenbank über StudIP (Einführung in Datenbanken → Informationen → Postgres-DB) ein und wählen Sie links das Schema `sakila_public`. Mit Link SQL-Command (links) kommen Sie zur Eingabe von SQL-Statements.

Actor (actor_id, first_name, last_name, last_update)

Address (address_id, address, address2°, district, city_id→City, postal_code°, phone, last_update)

Category (category_id, name, last_update)

City (city_id, city, country_id→Country, last_update)

Country (country_id, country, last_update)

Customer (customer_id, store_id→Store, first_name, last_name, email°, address_id→Address, active°, create_date, last_update°)

Film (film_id, title, description°, release_year°, language_id→Language, original_language_id° →language, rental_duration, rental_rate, length°, replacement_cost, rating°, special_features°, last_update)

Film_Actor (actor_id→Actor, film_id→Film, last_update)

Film_Category (film_id→Film, category_id→Category, last_update)

Film_Text (film_id, title, description)

Inventory (inventory_id, film_id→Film, store_id→Store, last_update)

Language (language_id, name, last_update)

Payment (payment_id, customer_id→Customer, staff_id→Staff, rental_id→Rental, amount, payment_date, last_update)

Rental (rental_id, rental_date, inventory_id→Inventory, customer_id→Customer, return_date°, staff_id→Staff, last_update)

Staff (staff_id, first_name, last_name, address_id→Address, picture°, email°, store_id→Store, active, username, password°, last_update)

Store (store_id, manager_staff_id→Staff, address_id→Address, last_update)

Geben Sie für die folgenden Anfragen entsprechende SQL-Statements an. Für jede Anfrage soll nur ein SQL-Statement angegeben werden. Falls die Tutoren eine SQL-Anfrage nicht schnell verstehen können, kann es Punktabzug geben. Sie können das

durch gute Formatierung und hilfreiche Kommentare (wo nötig) vermeiden. Beachten Sie aber, dass die Tutoren SQL natürlich sehr gut kennen. Vermeiden Sie triviale Kommentare. SQL-Statements mit Syntax-Fehlern werden mit Null Punkten bewertet. Geben Sie jede Teilaufgabe als separate Text-Datei mit der Endung .sql ab (kein Word). Unter Windows können Sie Text-Dateien mit Notepad editieren. Das Programm Notepad++ (`http://notepad-plus-plus.org`) bietet mehr Funktionen.

Beachten Sie, dass die SQL Anfragen keine Duplikate ausgeben sollen, d.h. verwenden Sie DISTINCT, wenn es notwendig ist. Für vergessenes oder überflüssiges DISTINCT werden Punkte abgezogen.

a) Geben Sie für jeden Kunden (`customer_id`, `first_name`, `last_name`) aus, wie oft er/sie Filme (`number_rentals`) aus einer Kategorie (`category_id, name`) ausgeliehen hat. Geben Sie die Kombination Kunde-Kategorie nur aus, wenn der Kunde in der Kategorie mehr als 5 Filme ausgeliehen hat. Mehrfaches Ausleihen eines Filmes zählt auch mehrfach. Sortieren Sie die Ausgabe erst nach `last_name`, dann nach `first_name` (beide alphabetisch aufsteigend) und abschließend nach Anzahl (`number_rentals`) (numerisch absteigend).

b) Gegeben Sie für jeden Kunden (`customer_id`, `first_name`, `last_name`) den oder die Lieblingsschauspieler (`actor_id`, `first_name`, `last_name`, `number_rentals`) aus. Der oder die Lieblingsschauspieler ist/sind diejenige(n), welcher in den meisten von dem Kunden ausgeliehenen Filmen mitspielte. Bei der Bestimmung der Anzahl der Filme sollen mehrfach ausgeliehene Filme auch mehrfach zählen.

c) Was haben die Filme (`film_id`, `title`, `ertrag`) bisher eingebracht? Summieren Sie die Beträge (`PAYMENT.AMOUNT`), die Kunden für einen Film bezahlt haben, um den Ertrag eines Filmes zu bestimmen. Sortieren Sie die Filme absteigend nach `ertrag` und dann aufsteigend nach `film_id`.