Der JOIN-Befehl

Begleitdokumentation

Fachdidaktik 1, GymInf
Januar 2023, David Tyndall

1 Einleitung

Dieses Unterrichtsmaterial soll in ca. 4-7 Lektionen Unterstützung bieten, um

- den `JOIN`-Befehl in seinen geläufigsten Varianten einzuführen,
- im Allgemeinen das Verteilen von Informationen auf mehrere Tabellen zu motivieren,
- zu verstehen, was eine Relation ist
- und das Abrufen der verteilten Information in SQL-Datenbanken einzuführen/zu üben.

Überblick Material

- Das Theorie-Unterrichtsmaterial für die Schülerinnen und Schüler (theorie.pdf). Die dort oft verwendete Restaurant-Datenbank ist unter restaurant.db (resp. restaurant.sql) gespeichert.
- Die Übungsunterlagen mit Aufgaben und Lösungen (uebungen.pdf). Die dort verwendete Datenbank ist unter uni.db (resp. uni.sql) gespeichert.
- Vorliegende Begleitdokumentation für die Lehrperson, inkl. Lösungen zu den Aufgaben im Theorie-Unterrichtsmaterial.

Notwendige Vorkenntnisse

Die Schülerinnen und Schüler

- wissen, dass in SQL die Informationen in Tabellen abgelegt werden. Die Spalten werden in den vorliegenden Unterlagen „Attribute“ und die Tupel „Zeilen“ genannt.
- wissen, dass jede Tabelle einen Primary Key hat. Dieser wird hier oft mit ID bezeichnet.
- kennen die Struktur `SELECT ... FROM ... WHERE`. In den Übungsaufgaben wird aktives Wissen vorausgesetzt, in der Theorie passives.
- verstehen, was `NULL` bedeutet.
- Falls das Kapitel „Relationen revisited“ behandelt wird, sollte der Begriff der *Funktion* aus der Mathematik bekannt sein.

Das Material wurde für Schülerinnen und Schüler im Alter von ca. 14-16 Jahren (also ca. 3.-4. Klasse von insgesamt 6 im Langgymnasium, resp. 1.-2. Klasse im Kurzgymnasium) entwickelt.

Zu erreichende Kompetenzen

- Die Schülerinnen und Schüler kennen die 4 verschiedenen Multiplizitäten von Relationen (1:1, 1:N, N:1, M:N). Sie können in einer gegebenen Situation aktiv erkennen, um welchen Typ es sich handelt. Ausserdem wissen sie, weshalb 1:N und N:1 gleichwertig sind. Die Schülerinnen und Schüler wissen, wie die verschiedenen Typen in SQL-Datenbanken gespeichert werden.
- Die Schülerinnen und Schüler wissen, wie in SQL über verschiedene Tabellen verteilte Informationen abgefragt werden können. Sie verstehen, weshalb die Informationen jeweils über die Primary Keys verbunden werden. Sie kennen die verschiedenen Varianten von JOIN (INNER, LEFT, RIGHT, OUTER, NATURAL) und wissen, was diese bewirken.
- Falls die Übungsaufgaben gelöst wurden: Die Schülerinnen und Schüler können aktiv SQL-Abfragen mit korrekt verwendeten JOINs schreiben.
- Die Schülerinnen und Schüler lernen, aufgrund des Resultates Rückschlüsse auf den Code zu machen.
- Die Schülerinnen und Schüler wissen, was eine Relation ist.
- Falls das Kapitel „Relationen revisited“ behandelt wird, ist den Schülerinnen und Schülern klar, dass diese eine Verallgemeinerung von Funktionen sind. Ausserdem kennen sie die Begriffe injektiv (linkseindeutig), surjektiv (rechtstotal), rechtseindeutig, linkstotal und bijektiv, und können Relationen korrekt nach diesen Eigenschaften kategorisieren.

2 Einsatz im Unterricht

Das Bearbeiten des Theoriematerials dauert ca. 4-5 Lektionen. Mit dem Lösen der Übungsaufgaben verlängert sich die benötigte Zeit um ca. 2 Lektionen.

Das Theorie-Unterrichtsmaterial kann ohne Computer, also insbesondere auch ohne Zugang zu einem Datenbanksystem bearbeitet werden. Die Teilaufgaben, bei denen ein solcher Zugang benötigt wird, sind mit 💻 gekennzeichnet. Diese können übersprungen werden, ohne dass das Verständnis der weiteren Inhalte beeinträchtigt wird. Bei den Übungsaufgaben ist ein DB-Zugang dagegen erforderlich.

Theorie-Material

Im Theorieteil sind einige Aufgaben eingebettet. Diese dienen eher der Erarbeitung des Stoffes und bieten nur wenig Vertiefung und Übung. Das Material kann zwar grundsätzlich von den Schülerinnen und Schülern selbstständig durchgearbeitet werden. Es ist aber dahingehend konzipiert, dass die Aufgaben im Plenum (oder wenigstens in Gruppen) besprochen werden, bevor man fortfährt.

In einigen der Aufgaben machen sich die Schülerinnen und Schüler selber Gedanken über die wesentlichen Ideen hinter den Konzepten, respektive sollen sie diese selbst finden. Ob sie nun auf die in SQL umgesetzten Lösungen stossen, oder ob sie Alternativvorschläge generieren, ist sekundär. Wichtiger ist, dass diese Ideen wohlwollend in der Klasse/Gruppe diskutiert werden. Vor- und Nachteile sollen erörtert werden. Dadurch werden die Ideen nicht nur besser in Erinnerung bleiben, es werden auch andere Kompetenzen weit über die Informatik hinaus trainiert (Dialektik, Darstellen von Information, etc.).

Da die Ideen der Aufgaben im Kapitel 2 („Beziehung zwischen Tabellen“) in den jeweils nachfolgenden Seiten vertieft werden, ist es empfehlenswert, die weiteren Seiten erst nach dem Lösen/Diskutieren der Aufgaben abzugeben/freizuschalten. Ausserdem macht es für die meisten Theorieaufgaben keinen Sinn, Musterlösungen an die Schülerinnen und Schüler abzugeben – wie oben erwähnt ist die Diskussion wichtiger.

In einigen anderen Aufgaben wird das Resultat einer SQL-Abfrage angegeben, und die Schülerinnen und Schüler sollen innerhalb eines beschränkten Rahmens herausfinden, was die Abfrage bewirkt. Neben dem offensichtlichen Ziel, eine neue Variante des Befehls kennen zu lernen, gibt es noch ein zweites: Die Fähigkeit, sich selbst Wissen durch das Studium von Code (ohne Erklärung) anzueignen, wird trainiert. Das dürfte beim Debuggen von Anfragen, und ganz generell beim Studium von Code hilfreich sein.

Der Theorieteil kann durchaus ohne die Übungsaufgaben und ohne Computer eingesetzt werden. Allerdings bleibt dann das Wissen über JOIN passiv. Man darf davon ausgehen, dass die Schülerinnen und Schüler Abfragen lesen können und verstehen, es ist jedoch nicht davon auszugehen, dass sie diese selbst schreiben können.

Das Kapitel „Relationen revisited“ kann problemlos übersprungen werden. Es ist als Vertiefung des Begriffs der Relationen gedacht und schlägt via Funktionsbegriff die Brücke zur Mathematik.

Übungsaufgaben

In den Übungsaufgaben werden die gelernten Konzepte vertieft und geübt. Ausserdem lernen die Schülerinnen und Schüler, eigenständig SQL-Abfragen mit den verschiedenen JOIN-Varianten zu schreiben. Mit den Aufgaben wird die aktive Kompetenz im Umgang mit JOIN gelernt (d. h. selber Abfragen schreiben können).

Falls im Unterricht bereits ein Datenbanksystem verwendet wird, ist es empfehlenswert, mit diesem weiterzuarbeiten. Falls noch keines vorhanden ist, sind im Übungsmaterial zwei Alternativen vorgeschlagen.

3 Lösungen zu den Aufgaben im Theorieteil

Aufgabe 1

a) Man könnte die Bestellung zum Beispiel durch Zuordnung der Zeilen mit Pfeilen darstellen. Vgl. Abschnitt „N:1-Beziehungen“.

b) Eine mögliche Darstellung könnte durch Erweitern der Tabelle um eine weitere Spalte erfolgen, in der das Menü angegeben wird. Vgl. Abschnitt „Speichern der Verbindungsinformation“.

c) Jede Person kann aus 5 Menüs auswählen. Da es vier Personen sind, gibt es $4 \cdot 5 = \underline{\underline{20}}$ Möglichkeiten.

Aufgabe 2

Andere Multiplizitäten von Relationen sind 1:1-, 1:N- und M:N-Beziehungen, vgl. nachfolgende Theorieseite.

Aufgabe 3

a) M:N-Beziehung. Jede Person, die das Restaurant besucht, kann mehrere Menüs mögen, und ein Menü kann von mehreren Besuchenden gemocht werden.

b) N:1-Beziehung. Jede Person, die das Restaurant besucht, hat maximal ein Lieblingsmenü. Dieses kann aber von mehreren Besuchenden bevorzugt werden.

c) 1:1-Beziehung. Jeder Teller kann maximal einmal genommen werden, und jeder Person steht maximal ein Teller zu (es ist ja schliesslich ein Hauptgang).

d) M:N-Beziehung. Beliebig vielen Besuchern können beliebig viele Menüs zu teuer erscheinen.

e) N:1-Beziehung. Gleiche Multiplizität wie bei den Bestellungen des aktuellen Restaurantbesuchs.

Aufgabe 4

- Daten-Integrität: Falls der Name des Menüs geändert wird, muss dies nur an einem Ort geschehen, und nicht etwa in jeder Personen-Zeile. So werden potenzielle Inkonsistenzen in den Tabellen verringert.
- Eindeutigkeit: Streng genommen müssen die Namen der Menüs nicht eindeutig sein (Spaghetti könnten beispielsweise mit verschiedenen Saucen angeboten werden). In der vorliegenden Tabelle wäre es allerdings kein Problem.

Aufgabe 5

a) Egal, die Fremdschlüssel können in der einen oder in der anderen Tabelle abgespeichert werden.

b) Das funktioniert nur in der zweiten Tabelle, sonst könnte es erforderlich werden, mehrere Fremdschlüssel in einem Feld der ersten Tabelle abzuspeichern.

c) Dies kann nur in der ersten Tabelle erfolgen, sonst könnte es erforderlich werden, mehrere Fremdschlüssel in einem Feld der zweiten Tabelle abzuspeichern.

Aufgabe 6

(a) Diese Relation erfüllt die Eigenschaften: injektiv (linkseindeutig), surjektiv (rechtstotal), 1:N-Beziehung. Es ist also keine Funktion.

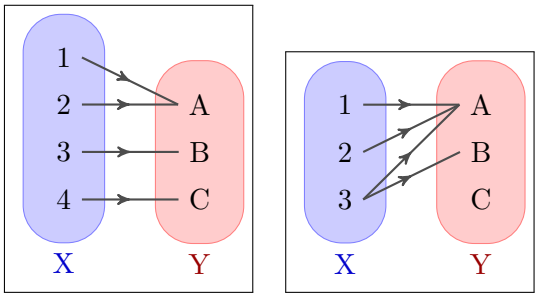
(b) Diese Relation erfüllt die Eigenschaften: injektiv (linkseindeutig), rechtseindeutig, linkstotal, surjektiv (rechtstotal), folglich bijektiv, 1:1-Beziehung.
Da die 1:1-Beziehung ein Spezialfall der N:1-Beziehung ist, und die Relation linkstotal sowie rechtseindeutig ist, handelt es sich um eine Funktion.

(c) Diese Relation erfüllt die Eigenschaften: injektiv (linkseindeutig), 1:N-Beziehung. Es ist also keine Funktion

(d) Diese Relation erfüllt die Eigenschaften: linkstotal, rechtseindeutig, N:1-Beziehung. Es ist also eine Funktion.

Aufgabe 7

Bei beiden Teilaufgaben gibt es viele korrekte Lösungen. Hier je ein Vorschlag:



a)

b)

Aufgabe 8

a) Jonas hat das Menü „Schnitzel mit Pommes“ bestellt (mit der ID 3).

b) Elias hat die ID 2. Wir betrachten also die Zeilen 3-5 in Tabelle 4. Dort finden wir die IDs 1, 2 und 3, was soviel wie Spaghetti, Ravioli und Schnitzel mit Pommes bedeutet.

c) Er hat etwas bestellt, das er mag: Jonas (ID=1) mag die Menüs mit ID 1 und 3. Bestellt hat er das Menü mit ID 3. (Hier ist es irrelevant, wie das Menü heisst.)

Aufgabe 9

Man erhält eine Tabelle, bei welcher die Zeilen zusammengefasst sind, welche die gleiche Personen- und Menü-ID haben. In diesem Kontext macht das nicht viel Sinn.

ID	Vorname	Nachname	MenuID	ID	Menu	Preis
1	Jonas	Allenmann	3	1	Spaghetti	21
2	Elias	Wirth	1	2	Ravioli	19
3	Jonas	Kopp	1	3	Schnitzel mit Pommes	23
4	Severin	Meier	NULL	4	Pilzrisotto	18

Aufgabe 10

a) In den Zeilen 3-4 gibt es ebenfalls eine Mehrdeutigkeit: Das Menü Spaghetti wird sowohl Elias Wirth als auch Jonas Kopp zugeordnet. Der Unterschied ist, dass bei dieser Mehrdeutigkeit einem Menu zwei Personen zugeordnet werden (in der ersten Mehrdeutigkeit war es genau umgekehrt). Allerdings ist in SQL die Reihenfolge der Tabellen bei JOIN irrelevant, daher handelt es sich nicht um einen prinzipiellen Unterschied, vgl. Aufgabe b).

b) Die Abfrage lautet dann:

```
1 SELECT *
2 FROM Personen AS p
3         JOIN Speisekarte AS s ON p.MenuID + 20 = s.Preis
```

Die Ausgabe unterscheidet sich nur in der Reihenfolge der Attribute; die ausgegebenen Zeilen sind ansonsten identisch. Hiermit wird nochmals unterstrichen, dass der in Aufgabe a) gefundene Unterschied nur eine Frage der Darstellung ist.

Dass die Reihenfolge kein prinzipieller Unterschied ist, kann mengentheoretisch begründet werden, SQL operiert ja immer auf Mengen. Ausserdem kann dies demonstriert werden, indem statt `*` in der `SELECT`-Klausel die Namen konkreter Attribute angegeben werden. Dann sind die beiden Ausgaben identisch.

Aufgabe 11

a) Bei Severin Meier hat das Attribut MenuID den Wert `NULL`. Dies kann mit keinem Menü verbunden werden. Infolgedessen wird diese Zeile nicht ausgegeben (nur die Schnittmenge wird ausgegeben).

b) Es wird jede Zeile aus der ersten Tabelle im Resultat erscheinen, egal, ob sie verbunden werden konnte. Falls nicht, werden die verschiedenen Attribute mit `NULL` aufgefüllt.

Aufgabe 12

a) Analog zum `LEFT JOIN`: Es werden alle Zeilen aus der zweiten Tabelle erscheinen, egal, ob sie mit Zeilen aus der ersten Tabelle verbunden worden sind. Leere Attribute werden mit `NULL` aufgefüllt.

Nachname	Menu
Wirth	Spaghetti
Kopp	Spaghetti
NULL	Ravioli
Allenmann	Schnitzel mit Pommes
NULL	Pilzrisotto
NULL	Cordon bleu

b) Man kann die 1. und 2. Tabelle einfach vertauschen. Dann hat der `LEFT-JOIN`-Befehl die gleiche Wirkung wie ein `RIGHT-JOIN`-Befehl bei der ursprünglichen Reihenfolge.

c) Die Abfrage mit der Variante `LEFT JOIN` lautet:

```
1 SELECT *
2 FROM Speisekarte AS s
3          LEFT JOIN Personen AS p ON p.MenuID = s.ID
```

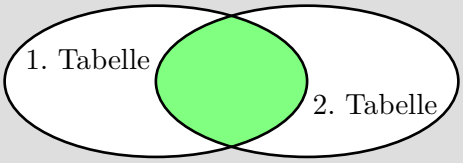
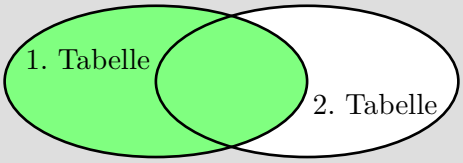
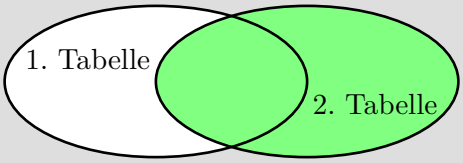
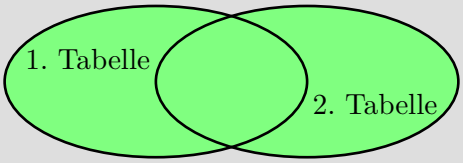
Aufgabe 13

a) `OUTER JOIN` gibt alle Zeilen aus, die in der ersten oder zweiten Tabelle erscheinen, egal, ob sie über die `ON`-Bedingung verbunden werden konnten. Wenn keine Zuordnungen zustande gekommen sind werden fehlende Attribute mit `NULL` aufgefüllt.

b) Wenn die `ON`-Bedingung fehlt, wird das kartesische Produkt aller Zeilen aus der ersten und zweiten Tabelle zurückgegeben. D. h., jede Zeile aus der ersten Tabelle wird mit jeder Zeile aus der zweiten Tabelle kombiniert. Dies ist die logische Fortsetzung davon, dass bei Mehrdeutigkeiten alle Kombinationen ausgegeben werden.

ID	Vorname	Nachname	MenuID	ID	Menu	Preis
1	Jonas	Allenmann	3	1	Spaghetti	21
1	Jonas	Allenmann	3	2	Ravioli	19
1	Jonas	Allenmann	3	3	Schnitzel mit Pommes	23
1	Jonas	Allenmann	3	4	Pilzrisotto	18
1	Jonas	Allenmann	3	5	Cordon bleu	23
2	Elias	Wirth	1	1	Spaghetti	21
2	Elias	Wirth	1	2	Ravioli	19
2	Elias	Wirth	1	3	Schnitzel mit Pommes	23
2	Elias	Wirth	1	4	Pilzrisotto	18
1	Elias	Wirth	1	5	Cordon bleu	23
3	Jonas	Kopp	1	1	Spaghetti	21
3	Jonas	Kopp	1	2	Ravioli	19
3	Jonas	Kopp	1	3	Schnitzel mit Pommes	23
3	Jonas	Kopp	1	4	Pilzrisotto	18
1	Jonas	Kopp	1	5	Cordon bleu	23
4	Severin	Meier	NULL	1	Spaghetti	21
4	Severin	Meier	NULL	2	Ravioli	19
4	Severin	Meier	NULL	3	Schnitzel mit Pommes	23
4	Severin	Meier	NULL	4	Pilzrisotto	18
1	Severin	Meier	NULL	5	Cordon bleu	23

Aufgabe 14

Befehl	Diagrammdarstellung
`JOIN` `INNER JOIN`	
`LEFT JOIN` `LEFT OUTER JOIN`	
`RIGHT JOIN` `RIGHT OUTER JOIN`	
`OUTER JOIN` `FULL OUTER JOIN`	

4 Quellen

Die Datenbank uni.db wurde leicht modifiziert aus der Gyminf-Vorlesung „Datenbanken“ von Prof. Dr. Sven Helmer, Universität Zürich übernommen.

Das Titelbild wurde am 20. November 2022 von `https://learn.microsoft.com/en-us/power-query/merge-queries-inner` entnommen.