Einführung in Datenbanken

Übung 10: NOT EXISTS, Nullwerte

Prof. Dr. Stefan Brass
PD Dr. Alexander Hinneburg
Martin-Luther-Universität Halle-Wittenberg

Wintersemester 2023/24

http://www.informatik.uni-halle.de/~brass/db23/

Inhalt

Haben Sie Fragen/Anmerkungen?

- Die Übung ist dafür da, dass Sie Fragen stellen können.
- Sie soll keineswegs ein Monolog werden.
- Haben Sie Fragen?
  - Organisatorisches?
  - Fragen zum Vorlesungsstoff?
- Haben Sie interessante neue Einsichten, die Sie teilen wollen?

Zur Bewertung der Klausur (1)

- Bei der Klausur gibt es pro SQL-Anfrage 5 Punkte.

  Das kann sich natürlich ändern.

- Sie müssen damit rechnen, dass ich für jeden Fehler einen Punkt abziehe (addiert sich bei mehreren Fehlern).
  - Bei Stilfehlern nur einen halben Punkt.
  - Wenn besonders wichtige Teile der Anfrage fehlen (z.B. ein ganzes `NOT EXISTS`) auch mehr als einen Punkt.
- Bei einem halben Punkt Abzug (90%) würde es vermutlich gerade noch für eine 1.0 reichen.

  Bei einem ganzen Punkt Abzug (80%) wäre man wohl an der unteren Grenze einer 1.7. Es kommt natürlich immer auf die Summe über alle Aufgaben an.

- Alle Angaben ohne Garantie!

  Man muss immer den Gesamtkontext der Aufgabe sehen.

Zur Bewertung der Klausur (2)

- Halber Punkt Abzug z.B. bei
  - `>` statt `>=`
  - Unnötiger Join
  - Überflüssiges `DISTINCT`
  - Verlangte Umbenennung einer Ausgabespalte fehlt.
  - Groß-/Kleinschreibung des Namens der Ausgabespalte nicht mit `"..."` geschützt.
  - `ORDER BY` fehlt oder falsch.
  - `ASC` bei `ORDER BY` nur in letzter Spalte angegeben

    Oder andere uneinheitliche Nutzungen von Defaults.
  - Wesentliche Codeduplizierung (Copy&Paste, z.B. bei `UNION`)

Zur Bewertung der Klausur (3)

- Weitere Beispiele für halben Punkt Abzug:
  - Logisch implizierte Bedingungen (überflüssig).
  - `IS NOT NULL` zusätzlich zum Test auf konkreten Wert

    Für `SELECT` bzw. `WHERE` is nur wichtig, ob die Bedingung wahr ist oder nicht. D.h. falsch oder unbekannt macht keinen Unterschied (nur, wenn es negiert wird).
  - `GROUP BY` statt `SELECT DISTINCT` (schlechter Stil)
  - Die gleiche Spalte zwei Mal im `GROUP BY` (überflüssig).
  - `COUNT(WNR)`, wenn nach `WNR` gruppiert wird.

    Es funktioniert, aber man zählt nur Duplikate (schlechter Stil).
  - Auffallend nichtssagende Namen (`hilf`, `temp`) für mehrere `WITH`-Hilfstabellen.

Zur Bewertung der Klausur (4)

- Weitere Beispiele für halben Punkt Abzug:
  - `LIKE` ohne Muster rechts (wo auch `=` ginge)
  - Eindeutig nicht portabele Konstrukte, z.B. `ILIKE`

    Es wird z.B. nicht in MariaDB und nicht in Oracle verstanden und war nicht in der Vorlesung dran. In der Vorlesung wurde gezeigt, wie man den case-insensitiven Vergleich mit `UPPER` macht.
  - Ebenso `ISNULL` statt `IS NULL` (deutlich nicht portabel)

    Ich wusste gar nicht, dass das geht, aber PostgreSQL akzeptiert es. Halten Sie sich an Konstrukte, die in der Vorlesung dran waren!
  - Komplizierte `SELECT`-Klausel in `EXISTS`-Unteranfrage
  - Zusätzliche Bedingung, die nicht in der Aufgabenstellung steht (aber auch nicht ganz sinnlos ist).

Zur Bewertung der Klausur (5)

- Ein Punkt Abzug z.B. bei
  - Einfacher Syntaxfehler

    Wenn Sie die Anfrage aufgrund des Syntaxfehlers nicht ausprobieren können, könnte es natürlich noch weitere Fehler geben, die auch Punkte kosten. Bei Syntaxfehlern bleibt es selten bei einem Punkt Abzug. Besonders schwere Syntaxfehler, die ein grundlegendes Unverständnis von SQL zeigen, können auch selbst mehr als einen Punkt kosten.
  - Unteranfrage als Term (z.B. unter `SELECT`) liefert mehr als eine Zeile (Laufzeit-Fehler)
  - Join-Bedingung fehlt
  - Notwendiger Join fehlt (z.B. ID aus Beispielzustand)
  - Klammern um `OR`-Bedingung vergessen
  - (Einfache) Bedingung der Aufgabenstellung fehlt

Zur Bewertung der Klausur (6)

- Weitere Beispiele für einen Punkt Abzug:
  - Falsches `GROUP BY`
  - `COUNT(DISTINCT WNR)` liefert immer 1, wenn nach `WNR` gruppiert wird.
  - Inkonsistente Bedingung (auch inkonsistenter Teil im `OR`)
  - Normaler Join, wo Outer Join gebraucht würde
- Beispiel für 1.5 Punkte Abzug:
  - Selbstverbund fehlt (wesentlicher Teil der Aufgabe)
- Beispiel für 2 Punkte Abzug:
  - `NOT EXISTS` fehlt

Inhalt

Beispiel/Aufgabe (1)

- Wählen Sie im Adminer das Schema „empdept_public“.
- Tabellen:
  - dept(deptno, dname, loc)
  - emp(empno, ename, job, mgr$^{\circ}\rightarrow$emp, hiredate, sal, comm$^{\circ}$, deptno$^{\circ}\rightarrow$dept)
- Schreiben Sie eine SQL-Anfrage, um leere Abteilungen zu finden, also Abteilungen ohne Angestellte.
- Erwartete Antwort:

deptno	dname
40	OPERATIONS

Die Lösung mit IN hat ein Problem, s.u.

Beispiel/Aufgabe (2)

- Mögliche Lösung:

```
SELECT d.deptno, d.dname
FROM   department d
WHERE  NOT EXISTS(SELECT *
                  FROM   emp e
                  WHERE  e.deptno = d.deptno)
```

Dies wäre ein „Antijoin“: Es gibt gerade keinen Verbundpartner in `emp`.

- Warum funktioniert folgende Lösung nicht?

```
SELECT deptno, dname
FROM   department
WHERE  deptno NOT IN (SELECT deptno
                      FROM   emp)
```

Erst Abstimmung, wer es weiss, dann Auflösung.

Beispiel/Aufgabe (3)

- Das Problem ist, dass es einen Angestellten mit einem Nullwert in der Spalte `deptno` gibt (`SMITH`).
- `IN` ist äquivalent zu `= ANY`, und das wird wie eine große Disjunktion ausgewertet.
- Für die Abteilung 40 ist die `NOT IN` Bedingung:

```
NOT(40 = 10 OR 40 = 20 OR 40 = 30 OR 40 = NULL)
```

  - Beim Vergleich mit dem Nullwert ergibt sich der dritte Wahrheitswert „unbekannt“.
  - Alle anderen Glieder der Disjunktion sind falsch.
  - Damit ist die Disjunktion „unbekannt“.
  - Die Negation von „unbekannt“ ist „unbekannt“.
  - Ausgaben gibt es nur, wenn die `WHERE`-Bedingung wahr ist.

Beispiel/Aufgabe (4)

- Lösung:

```
SELECT deptno, dname
FROM   department
WHERE  deptno NOT IN (SELECT deptno
                      FROM   emp
                      WHERE  deptno IS NOT NULL)
```

- Hier wird der Nullwert explizit ausgeschlossen aus der Wertemenge, die die Unteranfrage liefert.
- Hinweis: Der Test auf einen Nullwert muss mit `IS NULL` bzw. `IS NOT NULL` ausgeführt werden.

  `= NULL` bzw. `<> NULL` funktionieren nicht: Bei einigen Systemen (und nach dem Standard) ist das ein Syntaxfehler (Das Schlüsselwort `NULL` ist kein Term.). Bei anderen liefert es den dritten Wahrheitswert „unbekannt“ (Jeder normale Vergleich mit einem Nullwert liefert „unbekannt“.).

Inhalt

Präsenzaufgabe: Nichtmonotone Anfrage

- Tabellen des Schemas „empdept_public“ im Adminer:
  - dept(deptno, dname, loc)
  - emp(empno, ename, job, mgr$^{\circ}\rightarrow$emp, hiredate, sal, comm$^{\circ}$, deptno$^{\circ}\rightarrow$dept)
- Formulieren Sie die folgende Anfrage in SQL:
  - Gibt es einen Angestellten, der direkter Vorgesetzter (mgr) von allen Angestellten mit Job „SALESMAN“ ist (d.h. haben alle denselben Vorgesetzten)? Drucken Sie ggf. empno, ename, job:

7698	BLAKE	MANAGER

Falls es keinen Angestellten mit dieser Bedingung gibt, soll das Ergebnis der Anfrage leer sein.

- Verwenden Sie nur in der Vorlesung eingeführte Konstrukte.

Lösung der Präsenzaufgabe 6 (1)

- Gibt es einen Angestellten, der direkter Vorgesetzter (`mgr`) von allen Angestellten mit Job „`SALESMAN`“ ist (d.h. haben alle denselben Vorgesetzten)?

  Auch nichtmonoton: Wenn ich einen Salesman einfüge, dessen Vorgesetzter nicht Blake ist, kommt Blake nicht mehr heraus.

- Lösung 1:

```
SELECT empno, ename
FROM   emp e
WHERE  NOT EXISTS (SELECT *
                   FROM   emp x
                   WHERE  x.job = 'SALESMAN'
                   AND    x.mgr <> e.empno)
```

Lösung der Präsenzaufgabe 6 (2)

- Lösung 2:

```
SELECT empno, ename
FROM   emp
WHERE  empno = ALL (SELECT mgr
                    FROM   emp
                    WHERE  job = 'SALESMAN')
```

  Hier gibt es nun zwei Tupelvariablen, die beide `emp` heißen. Wenn man das stilistisch schlecht findet, kann man eine oder beide umbenennen.
  Da die Unteranfrage nicht korreliert ist, kann man sie aber getrennt entwickeln und dann nur in die äußere Anfrage „hineinstecken“.
  Insofern macht es durchaus Sinn, wie es ist.

- `ALL`-Bedingungen sind erfüllt, wenn die Unteranfrage ein leeres Ergebnis liefert.

Lösung der Präsenzaufgabe 6 (3)

Bemerkung zu Nullwerten:

- Im Prinzip kann die Spalte `mgr` einen Nullwert enthalten.
- Das kommt in den Daten nur für den `PRESIDENT` der Firma vor, nicht für einen `SALESMAN`.
- Aber das ist Wissen, was nicht in der Aufgabe steht, und nicht über einen `CHECK`-Constraint im Schema abgesichert ist.
- Streng genommen ist Lösung 1 falsch: Die Bedingung wäre auch erfüllt, wenn es einen `SALESMAN` gibt, der keinen Vorgesetzten hat (und es sonst nur einen Vorgesetzten aller Verkäufer gibt).

  Es gibt aber dennoch die volle Punktzahl für diese Lösung.
- Lösung 2 würde in dieser Situation das leere Ergebnis liefern.

Inhalt

Verschärfung der Regeln (1)

- Beachten Sie bitte, dass auch für schlechte Formatierung der Anfragen Punkte abgezogen werden können.

  Z.B. die ganze Anfrage in einer Zeile, wenn dabei 80 Zeichen überschritten werden. Sie müssen nicht die Formatierung wie auf den Folien wählen. Es ist nur gefordert, dass es übersichtlich aussieht (maximal 80 Zeichen pro Zeile, sowie eine gewisse Konsistenz und optische Unterstützung der syntaktischen Struktur wären wohl recht objektive Kriterien).

- Wählen Sie Bezeichner für Tupelvariablen, die den entsprechenden Tabellen leicht zugeordnet werden können.

  Bzw. die den beabsichtigten Inhalt der Zeile ausdrücken.

- Wenn Sie befürchten, dass Teile Ihrer Anfrage nicht unmittelbar einleuchtend sind, schreiben Sie Kommentare.

  Übertreiben Sie es nicht: Kommentare müssen einen Mehrwert haben, und nicht einzelne SQL-Konstrukte 1:1 in natürliche Sprache übersetzen.

Verschärfung der Regeln (2)

- Allgemein kann auch schlechter Stil einen Punkt kosten, zumindest soweit das bisher in Vorlesung und Übung thematisiert wurde, oder offensichtlich ist.
  - Das betrifft insbesondere die Verwendung von `LIKE` mit einer Zeichenkettenkonstante rechts ohne `%` und `_`.

    In diesem Fall würde `=` besser ausdrücken, was Sie meinen.
- Auch unnötige Verkomplizierungen der Anfrage können zu einem Punktabzug führen.
  - Z.B. unnötige Joins (Verbunde), also mehr Tupelvariablen unter `FROM` als nötig.

    Wenn Sie von einer Tupelvariable X ausschließlich den Schlüssel nutzen, und eine andere Tupelvariable Y einen Fremdschlüssel darauf enthält, und die entsprechende Verbund-Bedingung gefordert wird, wäre X überflüssig.

Schema der Vitamin-Datenbank

- Schema vit_public im Adminer:
  - Stoff_Kategorie(Kat, Bezeichnung, Sort_Nr)

    Neue Tabelle: Z.B. Vitamin, Mineralstoff, Spurenelement, ...
  - Stoff(Vit, Einheit, Tagesdosis°, Kat → Stoff_Kategorie)
  - Praeparat(Pid, Name, Hersteller, PZN°, Anz°, Einheit°, Tagesdosis°, Gewicht°, Preis°, glutenfrei°, lactosefrei°)
  - Inhalt(Pid → Praeparat, Vit → Stoff, Menge, Prozent°, Anmerkung°)
  - Zutat(Pid → Praeparat, Seq, Name, Anmerkung°)

Hausaufgabe 9.1: Selbstverbund (1)

- Welche Nahrungsergänzungsmittel (Präparate) enthalten
  - sowohl Calcium
  - als auch (Vitamin) K oder K2?
- Geben Sie Hersteller und Name des Präparats aus, sowie die Mengen an Calcium und Vitamin K/K_2.

  Sortieren Sie die Ausgabe nach der Menge an Calcium (größte Menge zuerst).
- Das Ergebnis sollte so aussehen:

Hersteller	Tabletten	Calcium	K
gesundleben	A-Z Vital	199.0	30.0
Pfizer	Centrum	162.0	30.0
Doppelherz aktiv	A-Z Complete	137.0	20.0
Abtei	A-Z Komplett	120.0	75.0

Hausaufgabe 9.1: Selbstverbund (2)

- Beispiel für eine SQL-Anfrage mit Selbstverbund.

  Man braucht für beide Vitamine jeweils eine Tupelvariable.

- Mögliche Lösung:

```
SELECT p.Hersteller AS "Hersteller",
       p.name AS "Tabletten",
       calcium.menge as "Calcium",
       k.menge AS "K"
FROM   praeparat p, inhalt calcium, inhalt k
WHERE  p.pid = calcium.pid
AND    p.pid = k.pid
AND    calcium.vit = 'Calcium'
AND    k.vit IN ('K', 'K2')
ORDER  BY "Calcium" DESC
```

Hausaufgabe 9.1: Selbstverbund (3)

- Was halten Sie von dieser Lösung?

```
SELECT P1.HERSTELLER AS "Hersteller",
       P1.NAME AS "Tabletten",
       I1.MENGE AS "Calcium", I2.Menge AS "K"
FROM   PRAEPARAT P1, PRAEPARAT P2,
       INHALT I1, INHALT I2,
       STOFF S2, STOFF_KATEGORIE SK2
WHERE  P1.PID = P2.PID AND P1.PID = I1.PID
AND    I1.VIT = 'Calcium'
AND    P2.PID = I2.PID AND I2.VIT = S2.VIT
AND    S2.VIT IN ('K', 'K2')
AND    S2.KAT = SK2.KAT
AND    SK2.BEZEICHNUNG = 'Vitamine'
ORDER  BY "Calcium" DESC
```

Hausaufgabe 9.2: NOT EXISTS (1)

- Welche Präparate enthalten zwar den Inhaltsstoff „Magnesium“, aber nicht die Zutat „Magnesiumoxid“?

  Geben Sie Hersteller und Name des Präparats sowie die Menge an Magnesium aus. Bei der Menge drucken Sie bitte in einer Spalte Zahlwert und Einheit, durch ein Leerzeichen getrennt (die Einheit finden Sie in der Tabelle `Stoff`).

- Das Ergebnis sollte so aussehen:

Hersteller	Name	Menge
Abtei	A-Z Komplett	56.0 mg
MensSana	Kardiodrink	112.5 mg
MensSana	Mineraldrink	375.0 mg
Verla-Pharm	Magnesium Verla 300	300.0 mg

Der letzte Hersteller ist eigentlich „Vela-Pharm Arzneimittel“.

Die Ausgabespalten sollen so heißen wie hier gezeigt. Sortieren Sie die Ausgabe nach Hersteller, bei gleichem Hersteller nach dem Namen des Präparats.

Hausaufgabe 9.2: NOT EXISTS (2)

- Mögliche Lösung:

```
SELECT p.Hersteller AS "Hersteller",
       p.name AS "Name",
       mag.menge || ' ' || s.einheit AS "Menge"
FROM   praeparat p, inhalt mag, stoff s
WHERE  p.pid = mag.pid
AND    mag.vit = 'Magnesium'
AND    s.vit = mag.vit
AND    NOT EXISTS
           (SELECT * FROM zutat z
            WHERE  z.pid = p.pid
            AND    z.name = 'Magnesiumoxid')
ORDER  BY "Hersteller", "Name"
```

Hausaufgabe 9.3: NOT EXISTS (1)

- Welche Inhaltsstoffe sind in dem Präparat „A-Z Vital“ enthalten, aber nicht in „Centrum“?
- Geben Sie den Namen des Stoffes aus und sortieren Sie danach.
- Das Ergebnis sollte so aussehen:

Inhaltsstoff
Lutein
Q10

Die Ausgabespalte soll so heißen wie hier gezeigt.

Hausaufgabe 9.3: NOT EXISTS (2)

- Mögliche Lösung:

```
SELECT in_vital.vit "Inhaltsstoff"
FROM   praeparat vital, inhalt in_vital
WHERE  vital.name = 'A-Z Vital'
AND    vital.pid = in_vital.pid
AND    NOT EXISTS
           (SELECT *
            FROM   praeparat centrum,
                   inhalt in_centrum
            WHERE  centrum.name = 'Centrum'
            AND    centrum.pid = in_centrum.pid
            AND    in_centrum.vit = in_vital.vit)
ORDER  BY "Inhaltsstoff"
```

Hausaufgabe 9.3: NOT EXISTS (3)

- Auch möglich (nicht äquivalent, aber Aufgabenstellung suggeriert, dass es Centrum in den Daten gibt):

```
SELECT in_vital.vit "Inhaltsstoff"
FROM   praeparat vital, praeparat centrum,
       inhalt in_vital
WHERE  vital.name = 'A-Z Vital'
AND    centrum.name = 'Centrum'
AND    in_vital.pid = vital.pid
AND    NOT EXISTS
           (SELECT *
            FROM   inhalt in_centrum
            WHERE  in_centrum.pid = centrum.pid
            AND    in_centrum.vit = in_vital.vit)
ORDER  BY "Inhaltsstoff"
```

Hausaufgabe 9.3: NOT EXISTS (4)

- Auch möglich (aber zusätzlicher Zugriff auf `Stoff`):

```
SELECT vit
FROM   stoff
WHERE  vit IN
       (SELECT vit FROM inhalt
        WHERE  pid = (SELECT pid
                      FROM   praeparat
                      WHERE  name = 'A-Z Vital'))
AND    vit NOT IN
       (SELECT vit FROM inhalt
        WHERE  pid = (SELECT pid
                      FROM   praeparat
                      WHERE  name = 'Centrum'))
ORDER  BY vit;
```

Hausaufgabe 9.4: IS NULL (1)

- Für welche Wirkstoffe (Tabelle `Stoff`) ist keine Tagesdosis definiert (d.h. es steht ein Nullwert in der Spalte `Tagesdosis`)?
- Geben Sie die Bezeichnung der Stoffkategorie und den Namen (`Vit`) des Stoffes aus, sowie die Sortier-Nummer der Stoff-Kategorie.
- Sortieren Sie nach dieser Nummer und innerhalb einer Kategorie nach dem Stoff-Namen.
- Die erwartete Antwort steht auf der nächsten Folie.

Hausaufgabe 9.4: IS NULL (2)

bezeichnung	vit	sort_nr
Mineralstoffe	Natrium	2
Carotinoide	Lutein	4
Carotinoide	Lycopin	4
Carotinoide	Zeaxanthin	4
Aminosäuren und -Verbindungen	L-Arginin	5
Aminosäuren und -Verbindungen	L-Carnitin	5
Sonstiges	Cholin	6
Sonstiges	Isoflavone	6
Sonstiges	Q10	6

Hausaufgabe 9.4: IS NULL (3)

- Eine mögliche Lösung ist:

```
SELECT k.Bezeichnung, s.vit, k.sort_nr
FROM   Stoff_Kategorie k, Stoff s
WHERE  s.kat = k.kat
AND    s.Tagesdosis IS NULL
ORDER  BY sort_nr, vit
```

Hausaufgabe 9.5: „Für alle“-Bedingung (1)

- Welche Stoffe sind in allen Präparaten enthalten, die „`A-Z`“ oder „`Centrum`“ als Teilstring in ihrem Namen haben?

  Es gibt im Prinzip auch Centrum-Präparate für spezielle Zielgruppen, z.B. „`Centrum FÜR SIE`“ und „`Centrum GENERATION 50+`“, deswegen könnte sich der Teilstring-Vergleich lohnen).

- Gesucht ist also die Schnittmenge der Wirkstoffe in allen diesen „Multivitamin“-Präparaten.

- Geben Sie den Kategorie-Namen mit aus und sortieren Sie nach der `Sort_Nr` der Kategorie und innerhalb der Kategorie nach dem Namen des Wirkstoffes (`Vit`).

- Die Ausgabe-Spalten sollen `Kategorie`, `Name` und `sort_nr` heißen.

Hausaufgabe 9.5: „Für alle“-Bedingung (2)

Kategorie	Name	sort_nr
Vitamine	A	1
Vitamine	B1/Thiamin	1
Vitamine	B12	1
Vitamine	B2/Riboflavin	1
Vitamine	B3/Niacin	1
Vitamine	B5/Pantothensäure	1
Vitamine	B6	1
Vitamine	B7/Biotin	1
Vitamine	B9/Folsäure	1
Vitamine	C	1
Vitamine	D	1
Vitamine	E	1
Vitamine	K	1
⋮	⋮	⋮
Spurenelemente	Zink	3

Hausaufgabe 9.5: „Für alle“-Bedingung (3)

```
SELECT k.Bezeichnung as "Kategorie",
       s.vit AS "Name", k.sort_nr
FROM   Stoff_Kategorie k, Stoff s
WHERE  s.kat = k.kat
AND    NOT EXISTS
       (SELECT *
        FROM   Praeparat p
        WHERE  (p.Name LIKE '%A-Z%'
                OR p.Name LIKE '%Centrum%')
        AND    NOT EXISTS(SELECT *
                          FROM   Inhalt i
                          WHERE  i.vit = s.vit
                          AND    i.pid = p.pid))
ORDER  BY sort_nr, "Name"
```