Restaurant - Die Datenbank

Ein Restaurant benoetigt eine Datenbank.

Beschreibung der Datenbank

Ein Restaurant wird eindeutig durch seine Restaurantnummer (RESTNR) und den Standort (STANDORT) identifiziert. Jedes Restaurant wird von genau einem Mitarbeiter geleitet, es können aber beliebig viele Mitarbeiter in einem oder mehreren Restaurants arbeiten. Ein Mitarbeiter wird eindeutig durch seine Personalnummer (PNR) identifiziert. Zusätzlich werden noch die Sozialversicherungsnummer (SVNR), sein Name (NAME), seine Anschrift (ANSCHRIFT) und das Eintrittsdatum (SEIT) gespeichert. Mitarbeiter können auch einen befristeten Arbeitsvertrag haben. Sollte dies der Fall sein wird zusätzlich noch das Ablaufdatum des befristeten Arbeitsvertrages (BIS) gespeichert. Neben allgemeinen Mitarbeitern gibt es auch noch Kellner. Jeder Kellner ist ein Mitarbeiter, es können aber nur Kellner für einen oder mehrere Tische zuständig sein. Aber für einen Tisch ist immer genau ein Kellner zuständig. Ein Tisch wird über seine Tischnummer (TISCHNR) eindeutig pro Restaurant identifiziert. Jedes Restaurant kann einen oder mehrere Tische enthalten. Auszerdem werden auch noch Bestellungen vom System erfasst. Eine Bestellung wird eindeutig über ihre Bestellnummer (BESTNR) identifiziert und ist immer genau einem Tisch zugeordnet. Pro Tisch kann es aber beliebig viele Bestellungen geben. Eine Bestellung besteht aus beliebig vielen Gerichten. Gerichte können in beliebig vielen Bestellungen zu einer bestimmten Stückzahl (STUECK) vorkommen. Ein Gericht wird eindeutig über seine Id (ID) identifiziert. Zusätzlich werden auch noch Name (NAME) und Preis (PREIS) des Gerichts vermerkt. Weiters werden auch Kunden in der Datenbank gespeichert. Ein Kunde wird über eine Id (ID) eindeutig identifiziert. Auszerdem werden noch der Name (NAME) und seine Telefonnummer (TELEFON) vermerkt. Ein Event wird eindeutig über sein Datum (DATUM) identifiziert. Auszerdem wird auch noch das Thema (THEMA) des jeweiligen Events gespeichert. An jedem Event können beliebig viele Kunden an beliebig vielen Tischen teilnehmen.

---

1)

Geben Sie alle KellnerInnen mit befristetem Dienstvertrag aus, deren Anschrift mit AUT endet. Sortieren Sie die Ergebnisse aufsteigend nach der SVNR.

```
SELECT * FROM MitarbeiterIn
WHERE pnr IN (SELECT pnr FROM Befristung)
AND pnr in (SELECT pnr FROM KellnerIn)
AND anschrift LIKE '%AUT'
ORDER BY svnr ASC;
```

```
SELECT MitarbeiterIn.*
FROM MitarbeiterIn JOIN Befristung ON MitarbeiterIn.pnr = Befristung.pnr
WHERE MitarbeiterIn.pnr in (SELECT pnr FROM KellnerIn)
AND anschrift LIKE '%AUT'
ORDER BY svnr ASC;
```

```
SELECT MitarbeiterIn.*
FROM MitarbeiterIn JOIN KellnerIn ON MitarbeiterIn.pnr = KellnerIn.pnr
WHERE MitarbeiterIn.pnr IN (SELECT pnr FROM Befristung)
AND anschrift LIKE '%AUT'
ORDER BY svnr ASC;
```

```
SELECT MitarbeiterIn.*
FROM MitarbeiterIn JOIN Befristung ON MitarbeiterIn.pnr = Befristung.pnr JOIN K
ellnerIn ON MitarbeiterIn.pnr = KellnerIn.pnr
WHERE anschrift LIKE '%AUT'
ORDER BY svnr ASC;
```

---

2)

Geben Sie alle Kunden (Name, Telefon) aus, welche im vor dem 06-06-2013 noch an keinem Event teilgenommen haben. Verwenden Sie hierzu die Funktion TO_DATE mit dem Datumsformat 'DD-MM-YYYY'. Sortieren Sie die Ergebnisse absteigend nach dem Namen.

```
SELECT name, telefon FROM Kunde
WHERE id NOT IN (SELECT id FROM teilnehmen WHERE datum < to_date('06-
06-2013', 'DD-MM-YYYY'))
ORDER BY name DESC;
```

```
SELECT name, telefon FROM Kunde
WHERE NOT EXISTS (SELECT id FROM teilnehmen WHERE datum < to_date(
'06-06-2013', 'DD-MM-YYYY') AND Kunde.id = teilnehmen.id)
ORDER BY name DESC;
```

3.1)

Geben Sie die Anzahl aller (unterschiedlichen) Events aus.

```
SELECT count(*) As anzahl
FROM Event;
```

3.2)

Geben Sie Kunden (ID) aus, die an Events teilgenommen haben, zusammen mit der Anzahl an (unterschiedlichen) Events, an denen Sie teilgenommen haben.

```
SELECT id, count(DISTINCT datum) AS anzahl
FROM teilnehmen
GROUP BY id;
```

3.3)

Geben Sie Kunden (Name, Telefon) aus, die an Events teilgenommen haben, zusammen mit der Anzahl an (unterschiedlichen) Events, an denen Sie teilgenommen haben.

```
SELECT Kunde.name, Kunde.telefon, count(DISTINCT datum) AS anzahl
FROM Kunde JOIN teilnehmen ON Kunde.id = teilnehmen.id
GROUP BY Kunde.id, Kunde.name, Kunde.telefon;
```

3.4)

Geben Sie jene Kunden (Name, Telefon) aus, die an den meisten (unterschiedlichen) Events teilgenommen haben.

```
SELECT Kunde.name, Kunde.telefon
FROM Kunde JOIN teilnehmen ON Kunde.id = teilnehmen.id
GROUP BY Kunde.id, Kunde.name, Kunde.telefon
HAVING count(DISTINCT datum) >= ALL(SELECT count(DISTINCT datum)
                                    FROM teilnehmen t
                                    GROUP BY t.id);
```

4.1)

Geben Sie das Eintrittsdatum der am laengsten beschaeftigten MitarbeiterInnen aus.

```
SELECT min(seit) AS seit
FROM MitarbeiterIn;
```

```
select seit
from mitarbeiterin m
group by seit
having seit <= all(
select seit from mitarbeiterin
);
```

```
SELECT distinct Seit
FROM Mitarbeiterin
WHERE Seit <= ALL(SELECT Seit
                  FROM Mitarbeiterin);
```

4.2)

Geben Sie jene MitarbeiterInnen (SVNR, Anschrift) aus, welche am laengsten beschaeftigt sind.

```
SELECT svnr, anschrift
FROM MitarbeiterIn
WHERE seit <= ALL(SELECT seit
                  FROM MitarbeiterIn);
```

```
SELECT svnr, anschrift
FROM MitarbeiterIn
WHERE seit = (SELECT min(seit)
              FROM MitarbeiterIn);
```

4.3)

Geben Sie jene MitarbeiterInnen (SVNR, Anschrift) aus, welche am laengsten beschaeftigt sind und in mindestens einem Restaurant arbeiten.

```
SELECT distinct svnr, anschrift
FROM MitarbeiterIn JOIN arbeitet ON MitarbeiterIn.pnr = arbeitet.pnr
WHERE seit <= ALL(SELECT seit
                  FROM MitarbeiterIn JOIN arbeitet ON MitarbeiterIn.pnr = arbeitet.pnr);
```

```
SELECT distinct svnr, anschrift
FROM MitarbeiterIn JOIN arbeitet ON MitarbeiterIn.pnr = arbeitet.pnr
WHERE seit = (SELECT MIN(seit)
        FROM MitarbeiterIn JOIN arbeitet ON MitarbeiterIn.pnr = arbeitet.pnr);
```

```
SELECT distinct svnr, anschrift
FROM MitarbeiterIn
WHERE pnr IN(SELECT pnr FROM arbeitet)
AND seit = (SELECT MIN(seit)
        FROM MitarbeiterIn
        WHERE pnr IN(SELECT pnr FROM arbeitet));
```

```
SELECT distinct svnr, anschrift
FROM MitarbeiterIn
WHERE pnr IN(SELECT pnr FROM arbeitet)
AND seit <= ALL(SELECT seit
            FROM MitarbeiterIn
            WHERE pnr IN(SELECT pnr FROM arbeitet));
```

---

5)

Geben Sie alle MitarbeiterInnen (svnr, anschrift) aus. In einer zusaetzlichen Spalte soll vermerkt werden, um welche Art von MitarbeiterIn es sich handelt, also entweder um eine 'KellnerIn' oder eine 'LeiterIn'. Sie koennen davon ausgehen, dass eine MitarbeiterIn entweder KellnerIn, LeiterIn oder keines von beiden ist. Sollte sie weder KellnerIn noch LeiterIn sein, soll in der zusaetzlichen Spalte 'Sonstiges' stehen.

```
        SELECT svnr, anschrift, 'KellnerIn' FROM KellnerIn JOIN MitarbeiterIn ON KellnerI
n.pnr = MitarbeiterIn.pnr
        UNION
        SELECT svnr, anschrift, 'LeiterIn' FROM MitarbeiterIn JOIN Restaurant ON pnr = l
eiter
        UNION
        SELECT svnr, anschrift, 'Sonstiges' From MitarbeiterIn WHERE pnr NOT IN (SEL
ECT pnr
                                                FROM KellnerIn
                                                UNION
                                                SELECT leiter FROM Restaurant);
```

---

6)

Geben Sie alle MitarbeiterInnen aus, welche in jedem Restaurant arbeiten.

```
SELECT pnr, name, svnr, anschrift, seit FROM MitarbeiterIn NATURAL JOIN arb
eitet
GROUP BY pnr, name, svnr, anschrift, seit
HAVING count(*) >= ALL(SELECT count(*)
        FROM Restaurant);
```

```
SELECT m.*
FROM MitarbeiterIn m
WHERE NOT EXISTS(
    SELECT *
    FROM Restaurant r
    WHERE NOT EXISTS(
        SELECT *
        FROM arbeitet a
        WHERE a.restnr = r.restnr AND a.standort = r.standort AND a.pnr = m.p
nr
    )
);
```

7.1)

Geben Sie zu den Restaurants (Standort, RestNr) in denen Bestellungen existieren den Gesamtumsatz (Summe der Preise aller bestellten Gerichte mal ihrer Stueckzahl) aus.

```
SELECT b.standort, b.restnr, sum(preis*stueck) AS tischsumme
FROM  Bestellung b JOIN beinhaltet i ON b.bestnr = i.bestnr
JOIN Gericht g ON i.id = g.id
GROUP BY b.standort, b.restnr;
```

7.2)

Geben Sie zu JEDEM Tisch (Standort, RestNr, TischNr) den Gesamtumsatz (Summe der Preise aller bestellten Gerichte mal ihrer Stueckzahl) aus. Tische ohne Bestellungen sollten also 0 Euro Umsatz gemacht

haben (Verwenden Sie dazu die Funktion COALESCE).

```
SELECT t.standort, t.restnr, t.tischnr, sum(COALESCE(preis*stueck, 0)) AS tisch
summe
FROM Tisch t LEFT OUTER JOIN Bestellung b
ON t.tischnr = b.tischnr AND t.standort = b.standort AND t.restnr = b.restnr
LEFT OUTER JOIN beinhaltet i ON b.bestnr = i.bestnr
LEFT OUTER JOIN Gericht g ON i.id = g.id
GROUP BY t.standort, t.restnr, t.tischnr;
```

```
SELECT t.standort, t.restnr, t.tischnr, COALESCE(
        (SELECT sum(stueck * preis)
         FROM bestellung best join beinhaltet bein on best.bestnr=bein.bestnr join
gericht ger on bein.id=ger.id
         WHERE best.standort=t.standort
         AND t.restnr=best.restnr
         AND t.tischnr=best.tischnr
         GROUP BY best.standort, best.restnr, best.tischnr), 0) as tischsumme
FROM tisch t;
```

8.1)

Geben Sie jene KellnerInnen (name, anschrift) aus, die den wenigsten Umsatz gemacht haben.

```
SELECT m.name, m.anschrift
FROM KellnerIn k JOIN MitarbeiterIn m ON (k.pnr=m.pnr) JOIN Tisch t ON (k.pnr =
 t.zustaendig) JOIN Bestellung b ON (t.restnr=b.restnr and t.standort=b.standort and t.tischnr=b.ti
schnr) JOIN beinhaltet be ON (b.bestnr=be.bestnr) JOIN Gericht g ON (be.id=g.id)
GROUP BY m.pnr, m.name, m.anschrift
HAVING sum(preis*stueck) <= ALL( SELECT sum(preis*stueck)
                                 FROM  Tisch
                                 NATURAL JOIN Bestellung NATURAL JOIN beinhaltet NATUR
AL JOIN Gericht
                                 GROUP BY zustaendig);
```

---

9)

Geben Sie eine Liste ALLER KellnerInnen (name) aus. Falls diese laenger als bis zum 06-06-2014 befristet sind, auch das Datum bis zu dem sie befristet sind. Ansonsten geben Sie das Defaultdatum 01-01-1999 aus. Verwenden Sie hierzu die Funktion TO_DATE mit dem Datumsformat 'DD-MM-YYYY'.

```
SELECT m.name, COALESCE(b.bis, TO_DATE('01-01-1999', 'DD-MM-YYYY'))
AS datum
 FROM (kellnerin k join mitarbeiterin m on(k.pnr=m.pnr)) left outer join befristung
b on (b.pnr=k.pnr and b.bis>TO_DATE('06-06-2014', 'DD-MM-YYYY'));
```

---

10)

Geben Sie die LeiterIn jener Restaurants (name) aus, zu denen Events existieren, dazu noch wieviele Kunden durchschnittlich an diesen Events teilgenommen haben.

```
SELECT m.name, avg(subt.teilnehmer) AS teilnehmer
FROM (
    SELECT standort, restnr, count(DISTINCT id) AS teilnehmer
    FROM teilnehmen NATURAL JOIN Restaurant
    GROUP BY standort, restnr, datum) subt
JOIN restaurant r on (subt.standort=r.standort and subt.restnr=r.restnr) JOIN Mitar
beiterIn m ON r.leiter = m.pnr
GROUP BY r.leiter, m.name;
```