Beantworten Sie die Fragen in Aufgabe 1 durch Ankreuzen; pro Frage genau ein Kreuz. Ist Ihre Antwort richtig, erhalten Sie einen Punkt. Ist sie falsch, wird Ihnen ein Punkt abgezogen. Machen Sie kein Kreuz, bleibt die Punktzahl unverändert. Sie erhalten mindestens 0 Punkte.

Aufgabe 1 (10 Punkte)

	Frage	**richtig**	**falsch**
1.	Die Attribute eines Fremdschlüssels dürfen unter keinen Umständen den Wert `null` annehmen.	○	⊗
2.	Die grundlegende Idee beim *offenen Hashing* ist es, die Records des Files auf mehrere *Buckets* aufzuteilen, die jeweils aus einer Folge von verzeigerten Blöcken bestehen.	⊗	○
3.	*Inkonsistenz* bedeutet, dass Informationen doppelt gespeichert werden.	○	⊗
4.	Persistente Daten sind Daten, die nach Beendigung des Programms nicht mehr existieren.	○	⊗
5.	ACID steht für Atomicity, Consistency, Isolation und Density.	○	⊗
6.	Das Zeitstempelverfahren dient zum Vermeiden von Deadlocks.	⊗	○
7.	*Lost Update*, *Dirty Write* und das *Phantomproblem* sind typische Fehler bei unkontrolliertem Mehrbenutzerbetrieb.	○	⊗
8.	Bei Verwendung eines *Typ-4*-JDBC-Treibers muß kein nativer Code auf dem Clientrechner installiert werden.	⊗	○
9.	Hintergundspeicherverlust tritt durch Absturz des DBMS ein.	○	⊗
10.	`grant insert on Vorlesungen to erika with grant option` bedeutet unter anderem, daß User "erika" anderen Usern Einfügerechte an der Tabelle "Vorlesungen" geben darf.	⊗	○

Aufgabe 2 (8 Punkte)

Erstellen Sie aus den folgenden Angaben ein ER-Diagramm, das einen Ausschnitt aus dem Online-Spiel Uga-Agga modelliert:

Entity-Typ	Attribute
Spieler	Loginname, Passwort
Goetter	Name, Element
Hoehlen	Id, Hoehlenname

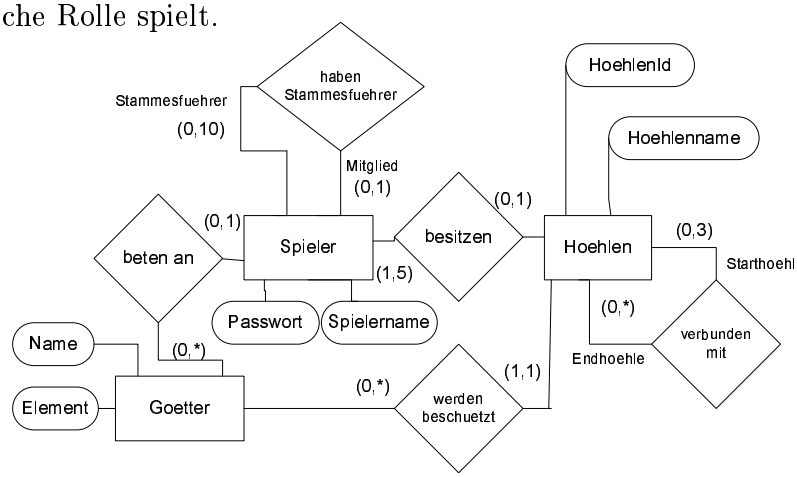
Relationship-Typ
besitzen
beten_an
verbunden_mit
haben_Stammesfuehrer
werden_beschuetzt

Geben Sie für die folgenden Beziehungen den Komplexitätsgrad in der (min, max)-Notation an:

1. Ein Spieler kann maximal fünf Höhlen besitzen.
2. Ein Spieler kann maximal einen Gott anbeten.
3. Ein Spieler hat höchstens einen Stammesführer.
4. Ein Stamm kann höchstens zehn Mitglieder haben.
5. Jeder Spieler besitzt mindestens eine Höhle.
6. Jede Höhle wird von genau einem Gott beschützt.
7. Es gibt mehr als fünfmal soviele Höhlen wie Spieler.
8. Es gibt weniger Götter als Höhlen.
9. Jeder Stammesführer hat sich selbst als Stammesführer.
10. Die Verbindungen zwischen den Höhlen sind *Einbahnstraßen*.
11. An einer Höhle beginnen höchstens drei Einbahnstraßen zu anderen Höhlen.
12. Jeder Spieler kann höchstens einen Stamm anführen.
13. Keine Höhle wird von zwei oder mehr Spielern besessen.
14. Eine Einbahnstraße verbindet genau zwei Höhlen.

Falls ein Entity-Typ an einer Relation mehr als einmal beteiligt ist, machen Sie kenntlich welches Entity welche Rolle spielt.

Lösung:



Aufgabe 3 (6 Punkte)

Das ER-Diagramm aus der letzten Aufgabe sei in folgendes relationale Schema überführt worden:

Spieler	:	{[Loginname, Passwort]}
Goetter	:	{[Name, Element]}
Hoehlen	:	{[HoehlenID,Hoehlenname]}
beten_an	:	{[Loginname,Name]}
besitzen	:	{[Loginname, HoehlenID]}
werden_beschuetzt	:	{[HoehlenID,Name]}
haben_Stammesführer	:	{[Loginname, Stammesführer]}
verbunden_mit	:	{[Starthoehle,Endhoehle]}

a) Welche Teile eines relationalen Schemas lassen sich theoretisch *verfeinern*? (2 Punkte)

Alle 1:N, 1:1 und N:1 Relationen können verfeinert werden, wenn Sie denselben Schluessel haben.

c) Verfeinern Sie das Schema aus a), wenn möglich und schreiben Sie es erneut auf. (4 Punkte)

Spieler	:	{[Loginname, Passwort, Gott, Stammesfuehrer]}
Goetter	:	{[Name, Element]}
Hoehlen	:	{[HoehlenID,Hoehlenname, Besitzer, Beschuetzer]}
verbunden_mit	:	{[Starthoehle,Endhoehle]}

Aufgabe 4 (6 Punkte)

Gegeben sei das unverfeinerte Schema aus Aufgabe 3.

a) Formulieren Sie die folgende Abfrage in der Relationenalgebra: (3 Punkte)
Wie lauten die Loginnamen der Spieler, die denselben Stammesführer haben wie 'Lehmann'?

$\Pi_{Loginname}(\sigma_{Stammesfuehrer=\Pi_{Stammesfuehrer}(\sigma_{loginname='Lehmann'}(haben\_Stammesfuehrer))}$
$(haben\_Stammesfuehrer))$
oder
$haben\_Stammesführer \div \Pi_{Stammesführer}(\rho_{Loginname='Lehmann'}(haben\_Stammesführer))$

b) Formulieren Sie die folgende Abfrage im relationalen Tupelkalkül: (3 Punkte)
Welche Höhlen werden von einem Gott mit dem Element `'Wasser'` beschützt?

$\{h|h \in Hoehlen \wedge \exists wb \in werden\_beschuetzt \wedge \exists g \in Goetter$
$\wedge wb.Name = g.Name \wedge g.Element =' Wasser' \wedge h.HoehlenID = wb.HoehlenID\}$

Aufgabe 5 (3 Punkte)

Gegeben sei das folgende SQL-Statement zum Anlegen der Tabelle `Goetter`:

```
CREATE TABLE Goetter ( Name VARCHAR(50) PRIMARY KEY, Element VARCHAR(50))
```

Ändern Sie das Statement so ab, daß das Attribut `Element` nur die Werte `'Erde'`, `'Feuer'`, `'Wasser'` und `'Luft'` annehmen kann und schreiben Sie das Statement erneut auf:

```
CREATE TABLE Goetter ( Name VARCHAR(50) PRIMARY KEY, Element VARCHAR(6)
CHECK (Element IN ('Erde', 'Feuer', 'Wasser', 'Luft')))
```

Aufgabe 6 (12 Punkte)

Gegeben sei das unverfeinerte relationale Schema aus Aufgabe 3. Formulieren Sie die folgenden Anfragen in SQL:

a) Wie lauten die Passwörter der Spieler, die den Gott `'Sirat'` anbeten? (2 Punkte)

```
SELECT Passwort from Spieler s, beten_an b
WHERE s.Loginname = b.Loginname
AND b.Name = 'Sirat'
```

b) Listen Sie die Namen der Spieler mit der Gesamtanzahl der Einbahnstraßen, die jeweils von all ihren Höhlen abgehen, in absteigender Reihenfolge. Namen von Spielern, von deren Höhlen keine Einbahnstraßen abgehen, müssen nicht mit aufgeltistet werden. (3 Punkte)

```
SELECT b.Loginname, count(b.HoehlenID) AS Einbahnstrassen
FROM besitzen b, verbunden_mit v
AND b.HoehlenID = v.Starthoehle
GROUP BY b.Loginname
ORDER BY Einbahnstrassen DESC
```

c) Wie lautet der Name des Gottes, der von den meisten Spielern angebetet wird? (4 Punkte)

```
SELECT Name  FROM (
  SELECT g.Name, count(*) AS Anzahl FROM Goetter g, beten_an b
  WHERE b.Name = g.Name
  GROUP BY g.Name) tmp
WHERE tmp.Anzahl >= all (
  SELECT count(*) FROM Goetter g, beten_an b
  WHERE b.Name = g.Name
  GROUP BY g.Name)
```

d) Welche Spieler haben keinen Stammesführer? (3 Punkte)

```
SELECT s.Loginname FROM Spieler s
WHERE NOT EXISTS (
  SELECT * FROM  haben_Stammesfuehrer h
  WHERE s.Loginname = h.Loginname)
```

oder (nur SQL 92)

```
(SELECT loginname FROM Spieler) MINUS
(SELECT loginname FROM haben_Stammesfuehrer)
```

Aufgabe 7 (6 Punkte)

Gegeben seien die folgenden Relationen (Schlüssel unterstrichen):

kunde : {[kd_nr, name, geb_dat]}
kauf : {[kd_nr, isbn_nr, datum]}
buch : {[isbn_nr, verlag, titel, sparte, preis]}

Wie lauten folgende SQL-Anfragen umgangssprachlich?

a) SQL-Anfrage:

```
SELECT DISTINCT name
FROM kunde, kauf
WHERE Datename(Day, kunde.geb_dat) = Datename(Day, kauf.datum)
AND Datename(Month, kunde.geb_dat) = Datename(Month, kauf.datum)
```

Umgangssprachlich (3 Punkte):

Wie heissen die Kunden, an deren Geburtstag ein Kauf getätigt wurde?

b) SQL-Anfrage:

```
SELECT DISTINCT kauf.kd_nr
FROM kauf, buch
AND kauf.isbn_nr = buch.isbn_nr
AND buch.preis = (SELECT MIN(preis) FROM buch)
```

Umgangssprachlich (3 Punkte):

Wie lauten die Kundennummern der Kunden, die das billigste Buch gekauft haben?

Aufgabe 8 (6 Punkte)

Gegeben seien die folgenden Relationen:

Vorlesungen: {[VorlNr: Integer, Titel: String, SWS: Integer]}

voraussetzen: {[Vorgänger: Integer, Nachfolger: Integer]}

a) Ein Student möchte die Vorlesung A hören. Um zu prüfen, ob er die nötigen Voraussetzungen mitbringt, möchte er eine SQL-Abfrage absetzen, die alle Vorgänger von A auflistet. Ist eine solche Abfrage in SQL möglich? Wenn ja: Wie lautet sie? Wenn nein: Warum nicht? (3 Punkte)

Das ist in SQL92 nicht möglich, weil dort keine Rekursionen möglich sind. Oracle:

```
select Titel from Vorlesungen where VorlNr in (select Vorgaenger
from voraussetzen connect by Nachfolger = prior Vorgaenger start with
Nachfolger = ( select VorlNr from Vorlesungen where Titel = 'A') )
```

b) In welchem logischen Datenmodell läßt sich eine solche Abfrage gut durchführen und warum? (3 Punkte)

Im OO-Modell ist das möglich, da jedes Objekt seinen Vorgänger und Nachfolger kennt, wenn die DB passend modelliert wurde.

Aufgabe 9 (4 Punkte)

Gegeben sei ein B^*-Baum mit k=3, der ein Datenfile mit 37 Datensätzen repräsentiert. Gesucht sei ein Schlüssel s, der nicht der kleinste Schlüssel im Baum ist.

a) Wieviele Baumknoten muß man in diesem Baum **mindestens** besuchen, um festzustellen, ob der Schlüssel s enthalten ist und ggf. die Adresse des Blockes mit dem Datensatz zu erhalten? (2 Punkte)

○ 2 Knoten ⊗ 3 Knoten ○ 5 Knoten ○ 7 Knoten

○ $\lceil \log_3 37 \rceil$ Knoten ○ $\lceil \log_2 37 \rceil$ Knoten ○ Alle Knoten ○ Keinen Knoten

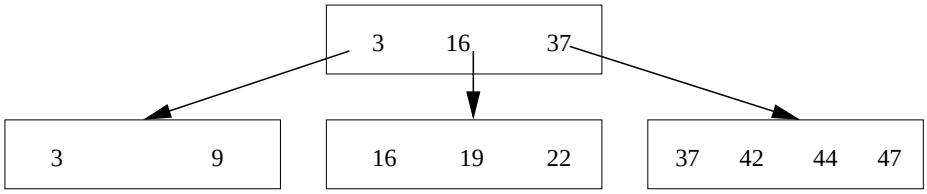
b) Wieviele Baumknoten muß man in diesem Baum **höchstens** besuchen, um festzustellen, ob der Schlüssel s enthalten ist und ggf. die Adresse des Blockes mit dem Datensatz zu erhalten? (2 Punkte)

○ 2 Knoten ⊗ 3 Knoten ○ 5 Knoten ○ 7 Knoten

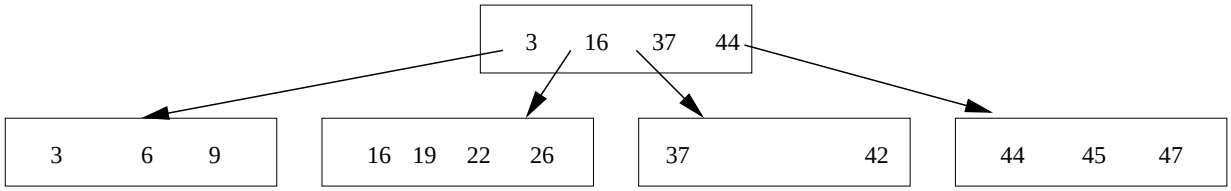
○ $\lceil \log_3 37 \rceil$ Knoten ○ $\lceil \log_2 37 \rceil$ Knoten ○ Alle Knoten ○ Keinen Knoten

Aufgabe 10 (8 Punkte)

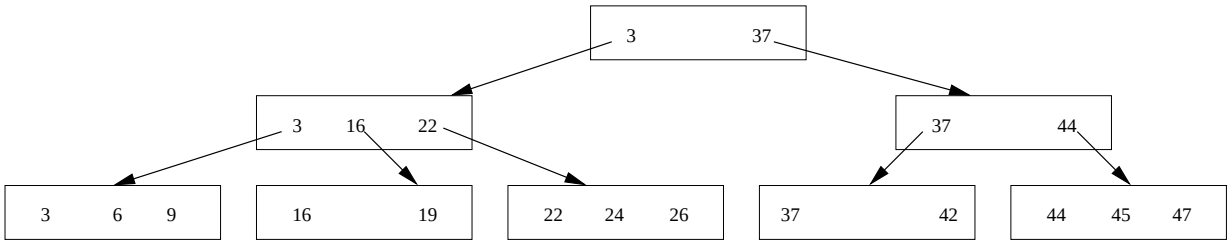
Zeichnen Sie den folgenden B^*-Baum (k=2) jeweils
1. nach dem Einfügen von 6, 26 und 45 (3 Punkte)
2. nach dem Einfügen von 24 (2 Punkte)
3. nach dem Löschen von 42 (3 Punkte)
einmal neu.



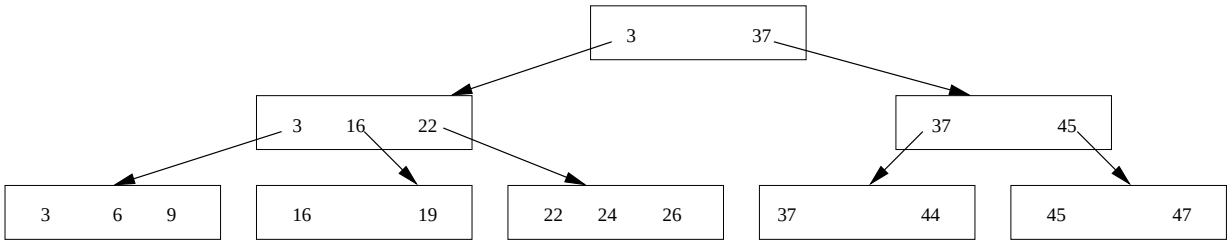
Nach dem Einfügen von 6, 26 und 45:



Nach dem Einfügen von 24:

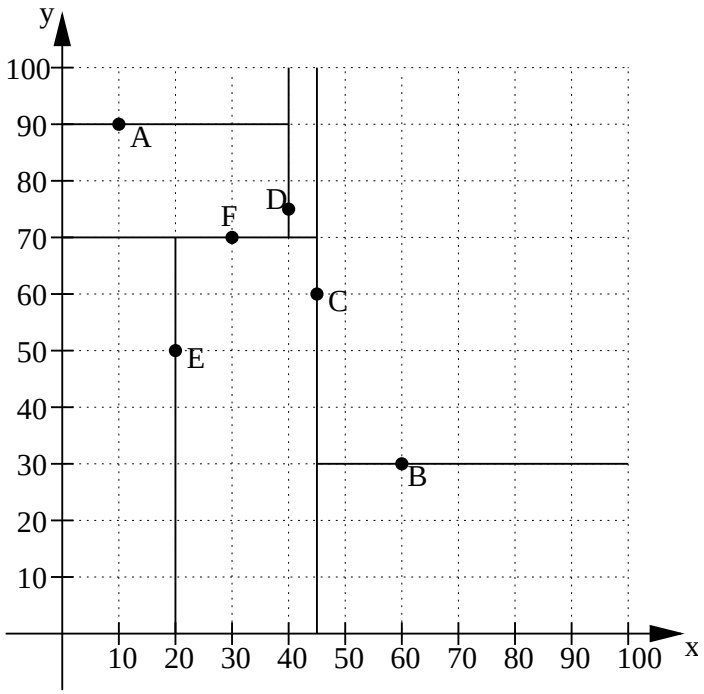


Nach dem Löschen von 42:

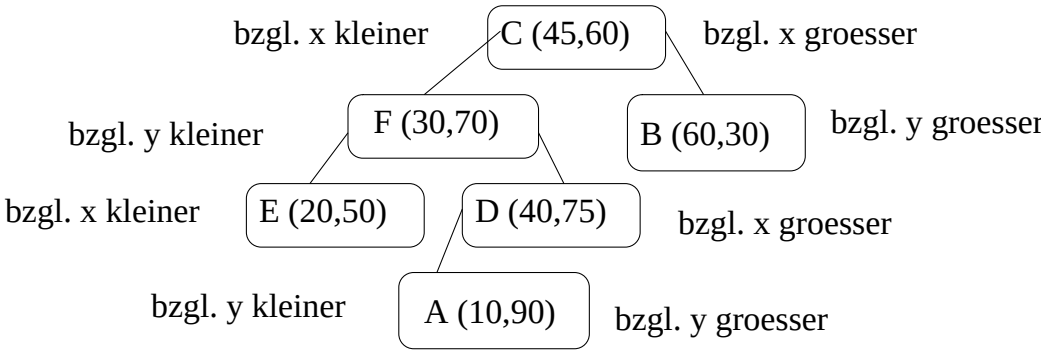


Aufgabe 11 (8 Punkte)

Gegeben sei folgende durch die Punkte $A = (10, 90), B = (60, 30), C = (45, 60), D = (40, 75), E = (20, 50)$ und $F = (30, 70)$ partitionierte Datenfläche:



a) Zeichnen Sie den zugehörigen 2-d-Baum mit den jeweiligen Buchstaben und Schlüsseln in den Knoten und den Diskriminierungsbedingungen. (6 Punkte)



b) Ist die Reihenfolge, in der die Datenrecords in den 2-d-Baum eingefügt wurden, durch die Partitionierung eindeutig festgelegt? (2 Punkte)

⊗Nein ○Ja

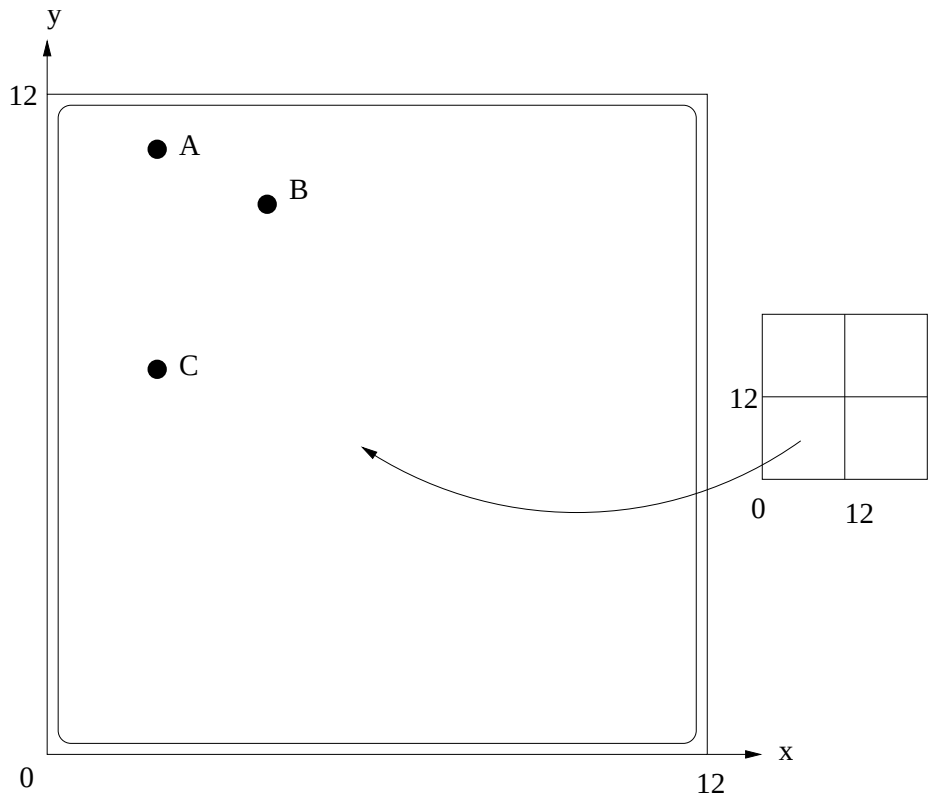
Aufgabe 12 (11 Punkte)

Gegeben sei ein Grid File mit Datenblöcken, die jeweils 3 Records aufnehmen können, und mit Directoryblöcken, die jeweils 4 Verweise auf Datenblöcke enthalten können.

Das zweidimensionale Schlüsseluniversum reicht in beiden Dimensionen von 0 bis 12.

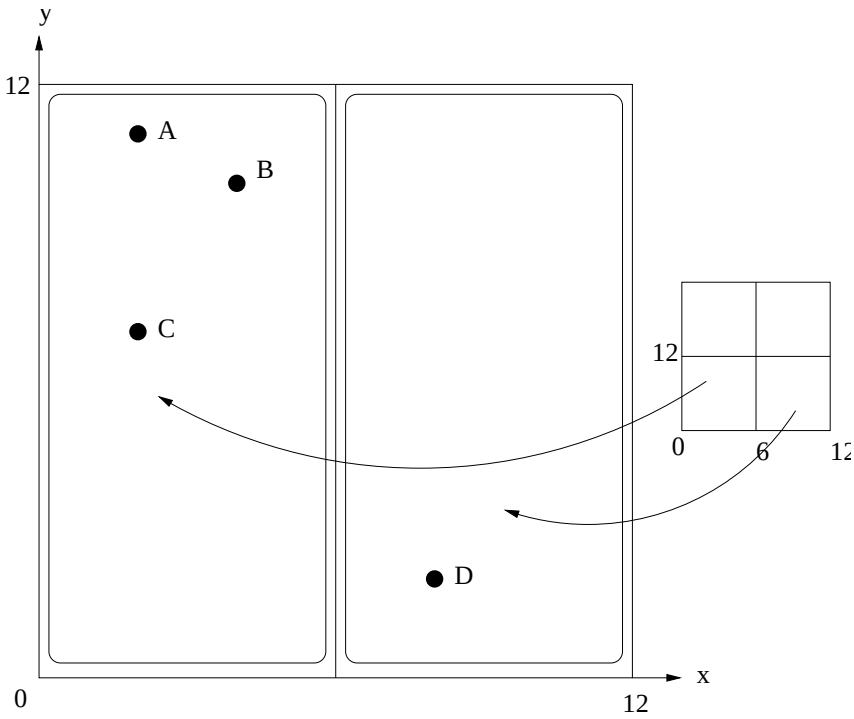
Das Aufsplitten der Regionen beginnt mit einer Halbierung der x-Achse.

Es wurden bereits 3 Datensätze in das Grid File eingefügt. Dadurch ergeben sich die Datenpunkte $A = (2, 11)$, ($B = (4, 10)$ und $C = (2, 7)$:

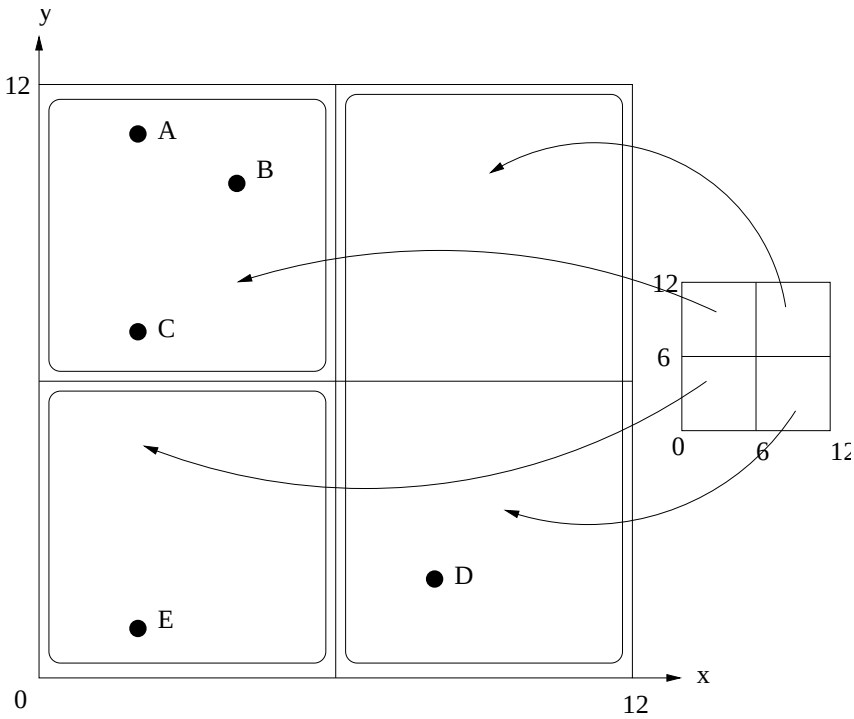


Aufgabenstellung: nächste Seite

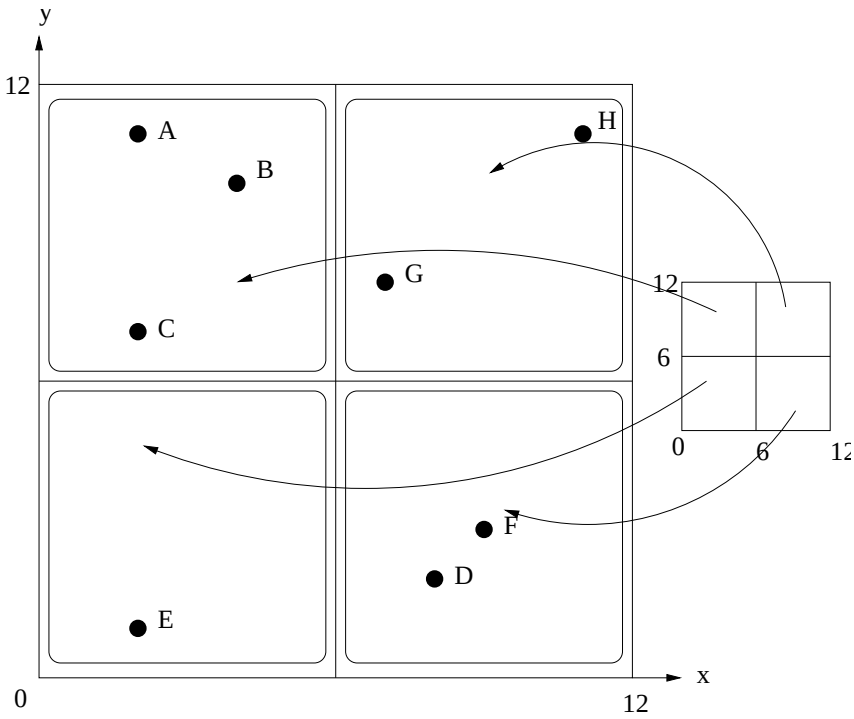
a) Fügen Sie den Punkt $D = (8, 2)$ ein und zeichnen Sie die Datenfläche und das Grid-Directory erneut. (2 Punkte)



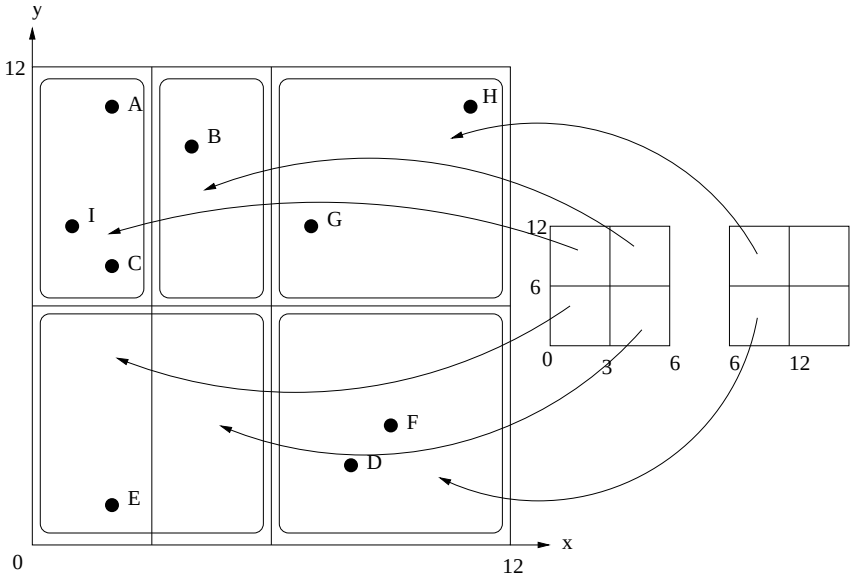
b) Fügen Sie den Punkt $E = (2, 1)$ ein und zeichnen Sie die Datenfläche und das Grid-Directory erneut. (2 Punkte)



c) Fügen Sie die Punkte $F = (9, 3)$, $G = (7, 8)$ und $H = (11, 11)$ ein und zeichnen Sie die Datenfläche und das Grid-Directory erneut. (4 Punkte)



d) Fügen Sie den Punkt $I = (1, 8)$ ein und zeichnen Sie die Datenfläche und das Grid-Directory erneut. (3 Punkte)



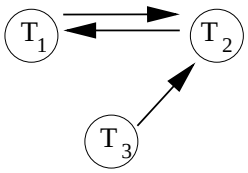
Aufgabe 13 (6 Punkte)

Betrachten Sie die drei Schedules mit den Transaktionen T_1, T_2, T_3:

Schedule 1	Schedule 2	Schedule 3
T_1: `BOT`	T_1: `BOT`	T_1: `BOT`
T_2: `BOT`	T_2: `BOT`	T_2: `BOT`
T_3: `BOT`	T_3: `BOT`	T_3: `BOT`
T_1: `lockX(c)`	T_1: `lockX(a)`	T_3: `lockX(b)`
T_2: `lockX(b)`	T_2: `lockX(a)`	T_2: `lockX(a)`
T_2: `lockX(c)`	T_3: `lockX(b)`	T_1: `lockX(c)`
T_3: `lockX(a)`	T_3: `lockX(a)`	T_1: `lockX(a)`
T_3: `lockX(b)`	T_1: `unlockX(a)`	T_3: `lockX(c)`
T_1: `lockX(b)`	T_1: `lockX(b)`	T_2: `lockX(b)`
...	...	...

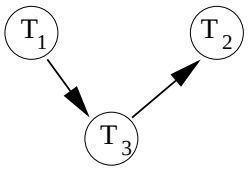
Zeichnen Sie den Wartegraphen zur Situation am Ende eines jeden Schedules. Liegt zu diesem Zeitpunkt ein Deadlock vor?

Schedule 1: Wartegraph:



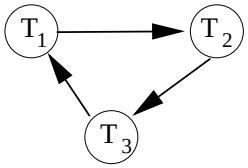
Deadlock? ⊗ Ja ○ Nein

Schedule 2: Wartegraph:



Deadlock? ○ Ja ⊗ Nein

Schedule 3: Wartegraph:



Deadlock? ⊗ Ja ○ Nein

Aufgabe 14 (6 Punkte)

Gegeben sei das Relationenschema
$R = \{A, B, C, D, E, F\}$
mit den funktionalen Abhängigkeiten

$$\begin{aligned} A &\rightarrow BC \\ E &\rightarrow ABC \\ F &\rightarrow CD \\ CD &\rightarrow BEF \end{aligned}$$

a) Nennen Sie alle Schlüsselkandidaten für R. (4 Punkte)

Es gibt zwei Schlüsselkandidaten:
F und CD
B steht niemals auf einer linken Seite. Es fallt daher aus. A impliziert nur außer B nur C, was nie alleine auf einer linken Seite steht. E impliziert neben A und B auch nur C.

b) R ist in der 1. Normalform. Ist R auch in der 2. Normalform? (2 Punkte)

- Nein, weil C von A, D und F voll funktional abhängig ist. ◯
- Ja, weil A, B und E von F und CD voll funktional abhängig sind. ⊗
- Nein, weil C von A u. D, aber auch von D u. F voll funktional abh. ist. ◯
- Ja, weil B kein Schlüsselkandidat ist. ◯
- Nein, weil F nur von E anhängig ist. ◯
- Ja, weil B und C von A und F voll funktional abhängig sind. ◯

Kreuzen Sie die richtige(n) Antwort(en) an.