	Lehrveranstaltung	Grundlagen von Datenbanken WS 2014/15		
	Aufgabenzettel	5		
	STiNE-Gruppe 30	Cornelia Hofsäß, Aleksej Davletcurin, Sascha Marcel Hacker		
	Ausgabe	Mi. 15.10.2014	Abgabe	Do. 31.10.2014

1 Referentielle Aktionen


a)

Bei einem bezüglich der referentiellen Aktionen sicheren Schemas ist das Ergebnis einer delete oder update Operation unabhängig von der Reihenfolge, in der sie durchgeführt wird.

b)

Im vorliegenden Schema gibt es zwei Fälle, in denen keine sichere referentielle Aktion durchgeführt werden kann.
Fall 1: Wenn eine Person gelöscht wird, welche einen Film ausgeliehen hat, wird die Operation unterbunden. Wenn aber zuerst die Filme gelöscht werden, von denen die Person Produzent ist und die Person genau die Filme, die er selbst produziert hat, ausgeliehen hat, wird die Operation nicht unterbunden. Je nach Reihenfolge der Löschoperation kommen unterschiedliche Ergebnisse heraus. Wir nehmen an, dass der ausgeliehene Film nicht im Bestand einer Videothek ist, ansonsten muss Fall 2 mitbeachtet werden.

Fall 2: Wenn der Produzent eines Filmes gleichzeitig der Leiter einer Videothek ist und dieser gelöscht wird, muss, damit die Löschoperation erfolgreich ist, zuerst die Videothek mit ihrem Bestand gelöscht werden, damit später der Film, den der Leiter der Videothek produziert hat, gelöscht werden kann. Wenn nachdem der Leiter gelöscht wurde, zuerst der Film gelöscht wird, dann wird die Löschoperation unterbunden, falls der Film im Bestand der Videothek ist.

	Lehrveranstaltung	Grundlagen von Datenbanken			WS 2014/15
	Aufgabenzettel	5			
	STiNE-Gruppe 30	Cornelia Hofsäß, Aleksej Davletcurin, Sascha Marcel Hacker			
	Ausgabe	Mi. 15.10.2014	Abgabe	Do. 31.10.2014	

2 Änderbarkeit von Sichten

a)

Listing 1: (i)-(iv)

```

1 create view FerariMechaniker (Vorname, Nachname) as
2 select Vorname, Nachname
3 from Mechaniker M, Rennwagen R
4 where M.wartet=R.RNr and R.Rennstall='Ferari';
5
6 create view reicheMechaniker(Vorname, Nachname) as
7 select Vorname, Nachname
8 from Mechaniker
9 where Gehalt > 2000000;
10
11 create view alteRennserien (Rennserie) as
12 select Rennserie
13 from Rennwagen
14 where Jahr < 1950;
15
16 create view Ferariwagen (RNr, Typ, Rennserie, Jahr) as
17 select RNr, Typ, Rennserie, Jahr
18 from Rennwagen
19 where Rennstall='Ferari';


```

Änderungsoperationen sind in SQL auf Tupel von Sichten erlaubt, wenn folgende drei Bedingungen erfüllt sind: Die Änderung in der Sicht muss eindeutig einem Tupel in der Basisrelation zugeordnet werden können. Es muss also in der Sicht ein Primärschlüssel beziehungsweise ein Schlüsselkandidat vorhanden sein. Zudem darf sich die Sicht nur auf eine einzige Tabelle beziehen und es dürfen in der Sicht keine Aggregatfunktionen, Gruppierungen oder Duplikateliminierung vorhanden sein, denn zum Beispiel kann nicht das Durchschnittsgehalt einer bestimmten Abteilung in der Basisrelation verändert werden.

Die Tupel in den Sichten FerariMechaniker und alteRennserien dürfen nicht verändert werden, da sich die Sicht FerariMechaniker auf zwei Tabellen bezieht und sich in der Sicht alteRennserien kein Primärschlüssel befindet. Dagegen dürfen die Tupel der Sichten reicheMechaniker und Ferariwagen verändert werden, da hier alle drei Bedingungen erfüllt sind: Die Sichten beziehen sich auf nur eine Tabelle, enthalten keine Aggregatfunktionen oder Gruppierungen und enthalten einen Primärschlüssel.

b)

Alle vier Sichten sind änderbar, da die drei oben genannten Bedingungen für alle Sichten erfüllt sind.

	Lehrveranstaltung	Grundlagen von Datenbanken WS 2014/15		
	Aufgabenzettel	5		
	STiNE-Gruppe 30	Cornelia Hofsäß, Aleksej Davletcurin, Sascha Marcel Hacker		
	Ausgabe	Mi. 15.10.2014	Abgabe	Do. 31.10.2014

Die erste Änderung bezieht sich auf die Sicht Formel1_Wagen. Durch die Angabe von 'WITH CASCADED CHECK OPTION' sind Änderungen nur zulässig, wenn das geänderte Tupel immer noch das Sicht Prädikat erfüllt, also das geänderte Tupel in der Sicht vorhanden bleibt. Dies ist hier der Fall, da nach der Änderung des Jahres das Tupel immer noch die Rennserie Formel 1 hat.

Diese Änderung ist nicht zulässig, da auch die Tabelle F156 die Spezifizierung 'WITH CASCADED CHECK OPTION' enthält, wodurch nur Änderungen zulässig sind, durch die die Sicht-Bedingung immer noch erfüllt bleibt. Wenn das Jahr auf 2014 gesetzt wird, dann fallen alle Tupel aus der Sicht F156 raus, da die Sicht nur Tupel enthält, bei denen das Jahr zwischen 1961 und 1963 liegt.

Diese Änderung ist zulässig. Durch die Änderung des Renstalls in 'Lotus' fliegen zwar alle Tupel aus der Sicht Ferrari_F1_Wagen heraus, jedoch bleiben alle Tupel in der Sicht Formel1Wagen enthalten, da das Attribut Rennserie nicht verändert wird.

Das Einfügen des Tupels ist nicht zulässig, da sich die Sicht Auto_Union_Rennwagen auf die Sicht Formel1_Wagen bezieht, und diese Sicht nur Rennwagen mit der Rennserie='Formel 1' enthalten darf und das Tupel, dass eingefügt wird zur Rennserie='Avus' gehört.

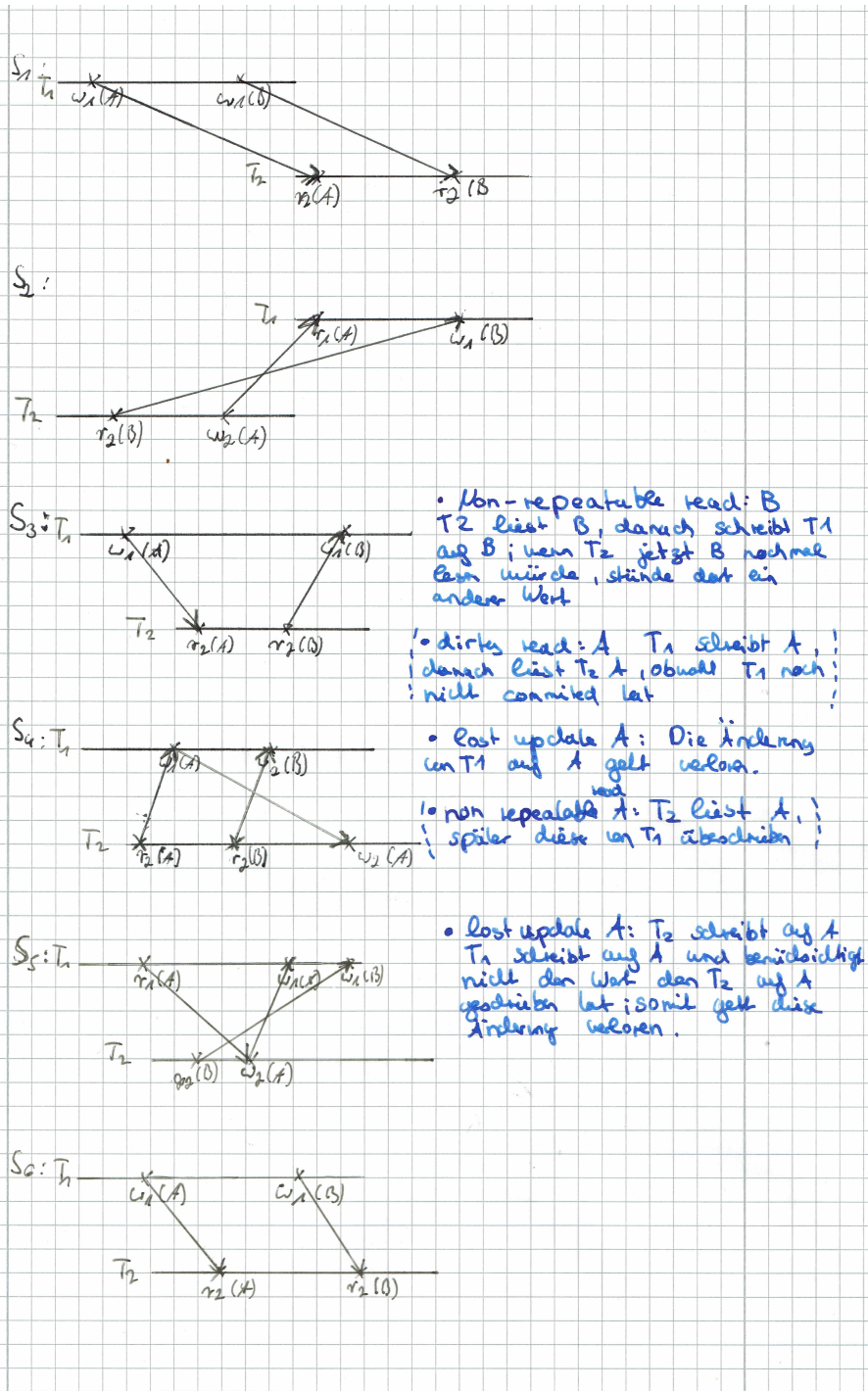
3 Serialisierbarkeit, Anomalien


a)

S_1 : A=320,B=10; S_2 : A=315,B=215; S_3 : A=315,B=10; S_4 : A=215,B=10; S_5 : A=110,B=10; S_6 : A=320,B=10;




b)



	Lehrveranstaltung	Grundlagen von Datenbanken WS 2014/15		
	Aufgabenzettel	5		
	STiNE-Gruppe 30	Cornelia Hofsäß, Aleksej Davletcurin, Sascha Marcel Hacker		
	Ausgabe	Mi. 15.10.2014	Abgabe	Do. 31.10.2014

c)

Der Schedule 1 und 2 sind seriell, da sie die Transaktionen direkt hintereinander ausführen. Die anderen Schedules sind nicht seriell. Zudem ist nur der Schedule 6 serialisierbar, da es zu der parallelen Ausführung eine serielle Ausführung gibt, die den selben Datenbankzustand und die selben Ausgabewerte erreicht. Datenanomalien und Begründungen sind in der Abbildung von Aufgabenteil b.

	Lehrveranstaltung	Grundlagen von Datenbanken			WS 2014/15
	Aufgabenzettel	5			
	STiNE-Gruppe 30	Cornelia Hofsäß, Aleksej Davletcurin, Sascha Marcel Hacker			
	Ausgabe	Mi. 15.10.2014	Abgabe	Do. 31.10.2014	

4 Transaktionen

	T ₁	T ₂	T ₃	
0				
1	lock(x, X)			
2	write(x)	lock(z, R)		
3		read(z)		
4	read(x)		lock(y, R)	
5		lock(y, X)	read(y)	T ₂ wartet auf T ₃
6	lock(y, X)			T ₁ wartet auf T ₃ und T ₂
7			write(y)	
8			unlock(y)	T ₃ weckt T ₂ auf
9		write(y)	commit	
10		read(z)		
11		unlock(y)		T ₂ weckt T ₁ auf
12	write(y)	unlock(z)		
13	unlock(y)	commit		
14	unlock(x)			
15	commit			
16				
17				
18				
19				
20				