Aufgabe 1

Beantworten Sie die folgenden Fragen und begründen oder erläutern Sie Ihre Antwort.

(a) Erläutern Sie die Begriffe Kardinalität und Partizipität. Welche Arten von Partizipität gibt es in der ER-Modellierung? Nennen und erklären Sie diese kurz.

Kardinalitäten Für die noch genauere Darstellung der Beziehungen im ER-Modell verwendet man Kardinalitäten (auch Grad der Beziehungen genannt). Diese geben an wie viele Entitätsinstanzen mit wie vielen Entitätsinstanzen einer anderen Entitätsinstanz in Beziehung stehen. [a]

Partizipation Die Partizipation eines Beziehungstyps (in einem Entity-Relationship-Modell) bestimmt, ob alle Entities eines beteiligten Entitätstyps an einer bestimmten Beziehung teilnehmen müssen. [b]

totale Partizipation: Wenn eine Beziehung Entität A und Entität B in Beziehung setzt, dann muss ein Eintrag in Entität A existieren, damit ein Eintrag in Entität B existiert und umgekehrt. Beide Entitäten müssen also an der Relation teilnehmen. Eine Entitätsinstanz aus A kann also nicht ohne eine in-Beziehung-stehende Entitätsinstanz aus B existieren und umgekehrt.

partielle Partizipation: Wenn eine Beziehung Entität A mit Entität B in Beziehung setzt, dann muss kein Eintrag in Entität A existieren, damit ein Eintrag in Entität B existieren kann und umgekehrt. Die beiden Entitäten müssen also nicht an der Relation teilnehmen (enthalten sein). [c]

[a] https://usehardware.de/datenbanksysteme-iv-entity-relationship-modell-er-modell-datenbankdarstellungen-i/
[b] https://lehrbuch-wirtschaftsinformatik.org/glossar/kapitel03/Partizipation
[c] https://usehardware.de/datenbanksysteme-iv-entity-relationship-modell-er-modell-datenbankdarstellungen-i/

(b) Mit welchen beiden Befehlen kann eine Transaktion beendet werden? Nennen Sie diese und erklären Sie den Unterschied.

Für den Abschluss einer Transaktion gibt es 2 Möglichkeiten:

- Den erfolgreichen Abschluss mit `commit`.
- Den erfolglosen Abschluss mit `abort`

(c) Erläutern Sie den Unterschied zwischen einer kurzen und einer langen Sperre.

lange Sperren: werden bis EOT gehalten (=> striktes 2PL)

kurze Sperren: werden nicht bis EOT gehalten

[a]

[a] https://www.dbs.ifi.lmu.de/Lehre/DBSII/SS2015/vorlesung/DBS2-03-Synchronisation.pdf

(d) Stellen Sie außerdem die Kompatibilitätsmatrix zur Umsetzung des ACID-Prinzips mit den richtigen Werten dar. S stehe dabei für eine Lese- und X für eine Schreibsperre.

	S	X
S	+	-
X	-	-

(e) Nennen und erklären Sie kurz die Armstrong-Axiome. Sind diese vollständig und korrekt?

Reflexivität: Eine Menge von Attributen bestimmt eindeutig die Werte einer Teilmenge dieser Attribute (triviale Abhängigkeit), das heißt, $\beta \subseteq \alpha \Rightarrow \alpha \rightarrow \beta$.

Verstärkung: Gilt $\alpha \rightarrow \beta$, so gilt auch $\alpha\gamma \rightarrow \beta\gamma$ für jede Menge von Attributen γ der Relation.

Transitivität: Gilt $\alpha \rightarrow \beta$ und $\beta \rightarrow \gamma$, so gilt auch $\alpha \rightarrow \gamma$.

Die Armstrong-Axiome sind korrekt und vollständig: Diese Regeln sind gültig (korrekt) und alle anderen gültigen Regeln können von diesen Regeln abgeleitet werden (vollständig). [a]

[a] https://dbresearch.uni-salzburg.at/teaching/2019ss/db1/db1_06-handout-1x1.pdf

(f) Was versteht man unter einem (Daten-)Katalog (Data Dictionary) und was enthält dieser (es genügt eine Auswahl zu nennen)?

Bei einer relationalen Datenbank ist ein Datenkatalog eine Menge von Tabellen und Ansichten, die bei Abfragen nur gelesen werden. Das Data-Dictionary ist wie eine Datenbank aufgebaut, enthält aber nicht Anwendungsdaten, sondern Metadaten, das heißt Daten, welche die Struktur der Anwendungsdaten beschreiben (und nicht den Inhalt selbst).

Zu einem Data-Dictionary zur physischen Datenmodellierung gehören genaue Angaben zu:

- Tabellen und Datenfeldern
- Primär- und Fremdschlüsselbeziehungen
- Integritätsbedingungen, z. B. Prüfinformationen

[a]

[a] https://de.wikipedia.org/wiki/Data-Dictionary

(g) Erklären Sie das konservative und das strikte Zwei-Phasen-Sperrprotokoll.

Konservatives 2-Phasen-Sperrprotokoll Das konservative 2-Phasen-Sperrprotokoll (Preclaiming), bei welchem zu Beginn der Transaktion alle benötigten Sperren auf einmal gesetzt werden. Dies verhindert in jedem Fall Deadlocks, führt aber auch zu einem hohen Verlust an Parallelität, da eine Transaktion ihre erste Operation erst dann ausführen kann, wenn sie alle Sperren erhalten hat.

Striktes 2-Phasen-Sperrprotokoll Das strikte 2-Phasen-Sperrprotokoll, bei welchem alle gesetzten Write-Locks erst am Ende der Transaktion (nach der letzten Operation) freigegeben werden. Dieses Vorgehen verhindert den Schneeballeffekt, also das kaskadierende Zurücksetzen von sich gegenseitig beeinflussenden Transaktionen. Der Nachteil ist, dass Sperren häufig viel länger gehalten werden als nötig und sich somit die Wartezeit von blockierten Transaktionen verlängert. Die Read-Locks werden entsprechend dem Standard-2PL-Verfahren entfernt.

(h) Erklären Sie die Begriffe „Steal/NoSteal“ und „Force/NoForce“ im Kontext der Systempufferverwaltung eines DBS.

No-Steal Schmutzige Seiten dürfen nicht aus dem Puffer entfernt und in die Datenbank übertragen werden, solange die Transaktion noch aktiv ist. Die Datenbank enthält keine Änderungen nicht-erfolgreicher Transaktionen. Eine UNDO-Recovery ist nicht erforderlich. langen Änderungs-Transaktionen können zu Problemen führen, da große Teile des Puffers blockiert werden

Steal Schmutzige Seiten dürfen jederzeit ersetzt und in die Datenbank eingebracht werden. Die Datenbank kann unbestätigte Änderungen enthalten. Eine UNDO-Recovery ist erforderlich. Es handelt sich um eine effektivere Puffernutzung bei langen Transaktionen mit vielen Änderungen.

Force Alle geänderten Seiten werden spätestens bei EOT (vor COMMIT) in die Datenbank geschrieben. Bei einem Systemfehler ist keine REDO-Recovery erforderlich. Die Force-Strategie benötigt einen hohen I/O-Aufwand, da Änderungen jeder Transaktion einzeln geschrieben werden. Die Vielzahl an Schreibvorgängen führt zu schlechteren Antwortzeiten, länger gehaltenen Sperren und damit zu mehr Sperrkonflikten. Große Datenbank-Puffer werden schlecht genutzt.

No-Force Änderungen können auch erst nach dem COMMIT in die Datenbank geschrieben werden. Die Änderungen durch mehrere Transaktionen werden „gesammelt“. Beim COMMIT werden lediglich REDO-Informationen in die Log-Datei geschrieben. Bei einem Systemfehler ist eine REDO-Recovery erforderlich. Die Änderungen auf einer Seite über mehrere Transaktionen hinweg können gesammelt werden.