NAME DES DOZENTEN: SCHULZ, WELTER

KLAUSUR: I110 - DATENBANKSYSTEME

QUARTAL: (1/2021)

Variante:

Dauer :90 Min

Datum: 2021-03-11

Seiten der Klausuraufgaben mit Deckblatt: 6
(hier bitte ggf. auch die Seitenanzahl der Anhänge wie Formelsammlungen mit aufführen)

Hilfsmittel: Es handelt sich um eine Open-Book-Klausur, d.h. es darf beliebiges Papier mit in die Prüfung genommen werden.

Bemerkungen:

- ***Bitte prüfen Sie zunächst die Klausur (alle Teile) auf Vollständigkeit***
- ***Bitte vermerken Sie auf Ihren Antwortbögen folgende Angaben:***
  - ***Name***
  - ***Matrikelnummer***
  - ***Zenturie***
  - ***Seitenzahl***

Es sind 100 Punkte erreichbar!
Zum Bestehen der Klausur sind 50 Punkte ausreichend!

Aufgabe	**Erreichbare Punkte**	**Erreichte Punkte**
Aufgabe 1: ER-Modellierung	16	
Aufgabe 2: Relationenmodell	12	
Aufgabe 3: Normalisierung	12	
Aufgabe 4: PL/SQL	12	
Aufgabe 5: SQL	24	
Aufgabe 6: Bewertungsfragen	24	
Summe		

Note:_____ Prozentsatz: _____ Ergänzungsprüfung:_____

Datum:_____ Unterschrift:_____

Datum:_____ Unterschrift:_____

Aufgabe 1: ER-Modellierung (16 Punkte)

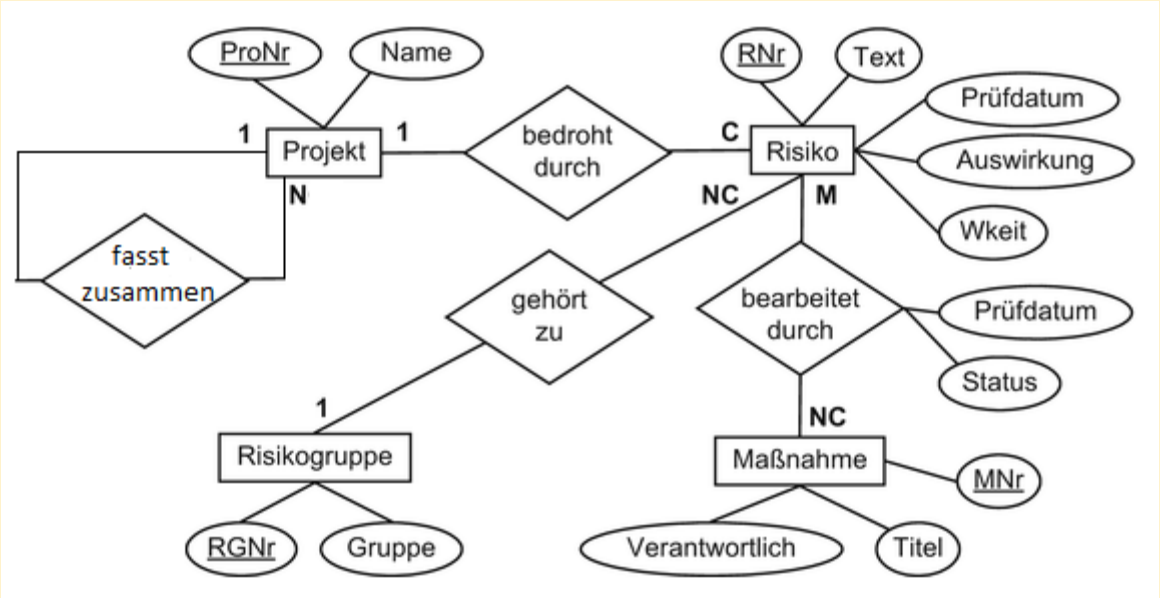
Einer Flughafen-Datenbank sollen folgende Fakten und Geschäftsregeln zu Grunde liegen:

(1) Jede Fluggesellschaft hat ein Kürzel, einen Namen und einen Stammsitz.
(2) Jedes Flugzeug hat eine Seriennummer, einen Hersteller, einen Typ und eine Passagierkapazität.
(3) Jeder Abflug hat eine Nummer, eine Abflugzeit (Datum und Uhrzeit) und einen Zielort.
(4) Jede Fluggesellschaft hat mindestens ein Flugzeug (seit bestimmtem Tag). Jedes Flugzeug wird von genau einer Fluggesellschaft betrieben.
(5) Für jeden Abflug wird genau ein Flugzeug eingesetzt. Flugzeuge sind (natürlich) wieder verwendbar.
(6) Jeder Flugzeugführer hat einen Namen und ein Geburtsjahr. Jede Fluggesellschaft hat mindestens einen zum Fliegen berechtigten Flugzeugführer.
(7) Für jeden Abflug braucht es zwei Piloten.

Entwickeln Sie für diese Anforderungen ein ER-Modell inklusive aller Schlüsselattribute und Kardinalitäten. Fügen Sie ggf. künstliche Schlüsselattribute hinzu, aber vermeiden Sie Redundanzen bzw. unnötige Schlüsselattribute.

Aufgabe 2: Relationenmodell (12 Punkte)

a. Leiten Sie aus folgendem Entity-Relationship-Modell das Relationenmodell ab (10 Punkte).



b. Beschreiben Sie textuell, welche Folgen es für das Relationenmodell hätte, wenn zwischen Projekt und Risiko eine 1:1-Beziehung modelliert worden wäre (2 Punkte).

Aufgabe 3: Normalisierung (12 Punkte)

In der Relation

LIEFERUNG (LieferNr, ArtikelNr, Artikelname, Anzahl, Lieferdatum,
Lieferfirma, AnschriftLieferfirma)

Bestehen folgende funktionale Abhängigkeiten zwischen den Attributen:

{LieferNr, ArtikelNr} → {Anzahl}
{LieferNr} → {Lieferdatum}
{ArtikelNr} → {Artikelname, Lieferfirma}
{Lieferfirma} → {AnschriftLieferfirma}

Weitere funktionale Abhängigkeiten lassen sich aus den angegebenen Abhängigkeiten berechnen. Sie können davon ausgehen, dass sich die Tabelle bereits mindestens in der 1. Normalform befindet.

a. Geben Sie die möglichen Schlüsselkandidaten für die Tabelle an (2 Punkte).
b. Formen Sie die Tabelle in Tabellen zweiter Normalform und dann in Tabellen dritter Normalform um, die die gleiche Aussagekraft haben. Kennzeichnen Sie dabei Primär- und Fremdschlüssel (10 Punkte).

Aufgabe 4: PL/SQL (12 Punkte)

Betrachtet wird ein Kartenspiel. Die Karten werden den Spielern zufällig zugeordnet, jeder Spieler erhält fünf bis zehn Karten. Das Ergebnis dieser Zuordnung wird in der Tabelle SPIEL gespeichert. Diese Tabelle ist folgendermaßen aufgebaut:

SPIELER_ID	FARBE	WERT
1	Karo	7
1	Pik	König
2	Kreuz	9
...	...	...

Erstellen Sie eine Funktion der eine SPIELER_ID übergeben werden kann und die Leerzeichen getrennt eine Zeichenkette mit sämtlichen Karo-Karten (Farbe und Wert), die diesem Spieler zugeordneten sind, zurückgibt. Die Karten sollen in alphabetischer Reihenfolge des Wertes ausgegeben werden. Gehen Sie vereinfachend davon aus, dass jeder Spieler mindestens eine Karo-Karte erhält.

Aufgabe 5: SQL (24 Punkte)

Die folgenden Tabellen werden von einer Autovermietung verwendet. Sie enthalten Informationen über die Standorte, an denen Kunden Fahrzeuge ausleihen können (z.B. Hamburg oder Berlin), Daten zu den Fahrzeugen und zu der Fahrzeugwartung.

```
CREATE TABLE Standort
(
  Stadt VARCHAR2(20) PRIMARY KEY,
  Parkplaetze NUMBER NOT NULL -- Anzahl der maximal verfügbaren Stellplätze am Standort
);

CREATE TABLE Fahrzeug
(
  Kennung NUMBER PRIMARY KEY,
  Kilometerstand NUMBER NOT NULL,
  Typ VARCHAR2(20) NOT NULL,
  Stadt VARCHAR(20),
  FOREIGN KEY (Stadt) REFERENCES Standort(Stadt)
);

CREATE TABLE Wartung
(
  Nummer NUMBER PRIMARY KEY,
  FahrzeugKennung NUMBER,
  Kosten NUMBER,
  Ergebnis VARCHAR(20),
  FOREIGN KEY (FahrzeugKennung) REFERENCES Fahrzeug(Kennung)
);
```

a. Skizzieren Sie die Tabellen, welche durch die DDL Statements erzeugt werden, inklusive Primär- und Fremdschlüssel. Es müssen keine Beispieldaten dargestellt werden. (3 Punkte)

Geben Sie bei den Aufgaben b bis e die DQL-Statements an, durch die die geforderten Daten abgefragt werden können. Die Punktzahl lässt keine Rückschlüsse auf die Komplexität der Antwort zu.

b. Geben Sie die Kennungen und Kilometerstände aller Fahrzeuge des Typs „SUV“ aus. (3 Punkte)

c. Geben Sie die Stadtnamen der Standorte in alphabetischer Reihenfolge aus. (3 Punkte)

d. Wie viele Parkplätze sind derzeit bei den einzelnen Standorten frei? (3 Punkte)

e. Geben Sie die Durchschnittskosten einer Wartung je Fahrzeugtyp an. (3 Punkte)

f. Über die Webseite der Autovermietung wird häufig angefragt, wie viele Fahrzeuge derzeit an einem Standort sind. Dabei dauert die SQL-Abfrage durch den JOIN über zwei Tabellen recht lange. Beschreiben Sie einen Ansatz, um die Situation zu entschärfen. (3 Punkte)

g. Die Einzigartigkeit einer Fahrzeugkennung ist sichergestellt, da es sich um einen Primärschlüssel handelt. Beschreiben Sie, wie stets eine neue, einzigartige Kennung für ein Fahrzeug generiert werden kann. (3 Punkte)

h. Erläutern Sie kurz für die folgenden SQL Statements, unter welchen Umständen diese erfolgreich ausgeführt werden können. (3 Punkte)

h1. INSERT INTO Standort VALUES ('Regensburg', 300);

h2. INSERT INTO Fahrzeug (Kennung, Kilometerstand) VALUES (45002, 100000);

h3. INSERT INTO Wartung VALUES (6, 130022, 301, 'Bestanden');

Aufgabe 6: Bewertungsfragen (24 Punkte)

Bitte bewerten Sie bei den nachfolgenden Aussagen, ob diese korrekt oder nicht korrekt sind. Begründen Sie Ihre Antwort jeweils kurz (je Aussage 2 Punkte).

a. Angenommen, ein Benutzer legt eine SEQUENCE in einem Oracle-Datenbanksystem mit dem folgenden Befehl an:

```
CREATE SEQUENCE myseq
INCREMENT BY 50
START WITH 100
MAXVALUE 2999
CYCLE;
```

Hier können demnach nur 58 verschiedene Werte automatisch erzeugt werden. Werden mehr benötigt, wird nach dem Wert 2950 der bereits verwendete Wert 50 noch einmal vergeben.

b. Die Verwendung der Klausel WITH CHECK OPTION beim Anlegen einer View führt dazu, dass möglicherweise INSERT-Statements abgelehnt werden, die auf der zugrunde liegenden Tabelle funktionieren würden.

c. Ein ROLLBACK zum letzten SAVEPOINT führt auf alle Fälle zu einem konsistenten Zustand der Datenbank.

d. Die Tabelle DUAL ermöglicht es beispielsweise, die aktuelle Systemzeit abzufragen.

e. Eine MATERIALIZED VIEW ist eine Kopie der zugrundeliegenden Tabelle(n) und wird wie eine „normale“ View benutzt. Der Hauptunterschied ist, dass die Performanz deutlich besser ist, aber Datenänderungen nachgeführt werden müssen.

f. Dieses SQL-Statement ändert bei einer Tabelle den Namen der Spalte „Gehalt“:

```
MODIFY TABLE RENAME COLUMN Gehalt TO 'Lohn';
```

g. Bei der Verwendung der folgenden VIEW-Definition führt der angegebene INSERT-Befehl zu einer Fehlermeldung.

```
CREATE VIEW DBZenturie AS
SELECT S.Name, S.Studiengang, S.Jahrgang
FROM Studierende S JOIN Module M ON S.Zenturie = M.Zenturie
WHERE M.Name = 'I110' and S.Jahrgang = 2019
        WITH CHECK OPTION;
        INSERT INTO DBZenturie VALUES ('Herbert Mustermann','AInf','2020');
```

h. Damit ein Schema in dritter Normalform (3NF) ist, genügt es zu zeigen, dass keine transitiven Abhängigkeiten zwischen Nicht-Schlüsselattributen bestehen bzw. durch Zerlegung in mehrere Tabellen aufgelöst wurden.

i. Die folgende Trigger-Definition ist fehlerfrei.

```
CREATE OR REPLACE TRIGGER ueberweisungslogger
  BEFORE INSERT ON ueberweisungen
  DECLARE
    datum DATE;
  BEGIN
    SELECT SYSDATE INTO datum FROM DUAL;
      INSERT INTO mylogfile
        VALUES(datum,:NEW.betrag, :NEW.sender, :NEW.empfaenger)
  END;
```

j. Die folgende Funktions-Definition ist fehlerfrei.

```
CREATE OR REPLACE FUNCTION ist_dispo_ok
  (betrag INTEGER, kontostand INTEGER, dispo INTEGER)
  RETURN BOOLEAN IS ergebnis INTEGER DEFAULT 0;
  DECLARE
    zwischensumme INTEGER;
          BEGIN
            zwischensumme = kontostand - betrag;
            IF zwischensumme < dispo THEN RETURN FALSE;
            ELSE RETURN TRUE;
            END IF;
          END;
```

k. Um mit einem CURSOR zu arbeiten, muss der Cursor zunächst definiert, dann mit OPEN zum ersten Mal geöffnet, zeilenweise per FETCH ausgelesen und am Schluss per CLOSE geschlossen werden. Die Aufrufe von OPEN, FETCH und CLOSE kann man sich aber in FOR-Loops sparen.

l. Im Gegensatz zu einem Trigger hat eine Prozedur einen Rückgabewert.