Aufgabe 1: Anforderungsanalyse und Datenmodellierung (8 Punkte)

Die Dirtbike-Rental-GmbH verleiht Motorräder für europäische und afrikanische Abenteuerurlaube. Es gibt in Europa und Afrika Verleihbüros. Unser Verleihbestand umfasst insgesamt über tausend Fahrzeuge. Benötigt wird ein System, mit dem Verleihverträge gespeichert werden können.

Jedes Büro verleiht die verfügbaren Fahrzeuge an Kunden. Die Rückgabe erfolt in der Regel bei einem anderen Büro. Wir nehmen keine Reservierungen entgegen. Jedes Verleihbüro hat einen Namen und eine eindeutige Nummer, sowie eine Adresse. Jedes Büro verwaltet einige Fahrzeuge, umgekehrt gehört jedes Fahrzeug einem Büro. Jedes Fahrzeug verfügt über eine ID, einen Zulassungsort und eine Zulassungsnummer.

Wichtig ist das Datum der letzten Inspektion und das Datum an dem die Zulassung abläuft. Wir brauchen den Namen, die Telefonnummer, die Adresse eine oder mehrere Kontoverbindungen und die Führerscheindaten des Ausleihenden. Wir möchten über alle unsere Kunden Buch führen, und ihn ggf. als Risikokunden einstufen.

Für jedes Fahrzeug wird ein Vertrag abgeschlossen. Meist leihen Kunden zwei oder mehr Fahrzeuge gleichzeitig aus. Jeder Verleihvertrag ist durch die Nummer des Büros und eine Vertragsnummer gekennzeichnet. Wichtig sind auch folgende Informationen:

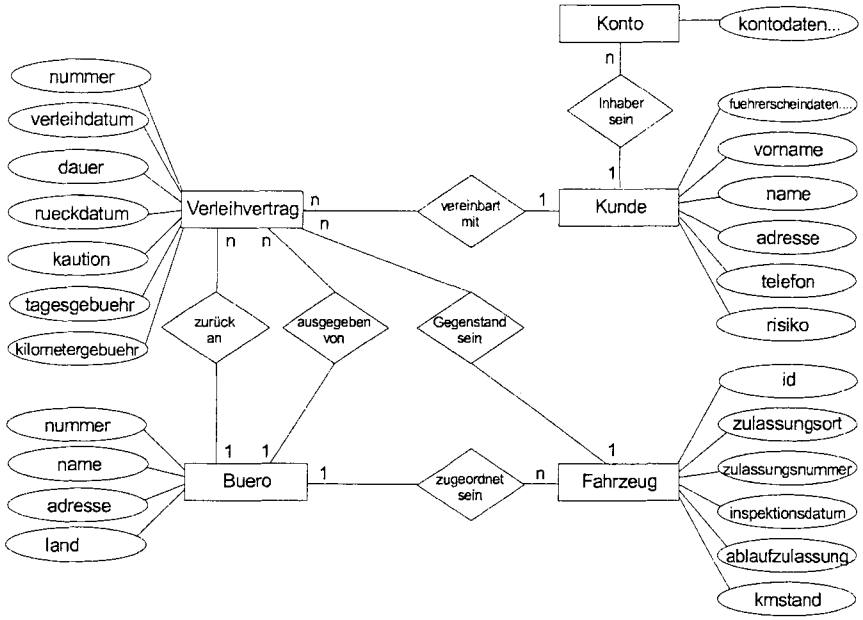
- Verleihdatum
- angestrebte Verleihdauer
- ursprüngliches Verleihbüro
- das Büro, bei dem das Fahrzeug abgegeben wird
- die hinterlegte Kaution
- den ausgehandelten Mietpreis pro Tag
- die ausgehandelte Kilometergebühr

Die finanzielle Seite des Geschäftes wird zunächst nicht automatisiert, sondern nur die Verfolgung der Verleihverträge und die Zuordnung der Fahrzeuge.

Entitytypen Attribute

Jeweils ½ Punkt – aufrunden

Aufgabe 2: Entity-Relationship-Diagramm (15 Punkte)



Anhaltspunkt: ½ Punkt pro Item (wären insg. 17,5), Abzüge für Foreign Keys im Diagramm.

Aufgabe 3: Datenmodellierung und Schemamanipulation (15 + 5 Punkte)

Hinweis: Die Foreign Key Constraints können natürlich auch in die CREATE TABLE Statements integriert werden.
a) 5 Punkte Grau: +3 Punkte
Kopie (id, einkaufspreis, standort, defekt, anzahldvds)
Titel (id, titel, datum, fskfreigabe, kritik, tit_typ, fil_regisseur, fil_extras, fil_genre, gam_firma, gam_plattform, gam_kategorie, preisstufe_id)
Titel_Kopie (titel_id, kopie_id)
Preisstufe (id, gebuehr, notiz)
Vormerkung (id, abgelaufen, benachrichtigt, kopie_id, titel_id)

b) Pro CREATE TABLE mit dazugehörigen Foreign Keys (ALTER TABLE oder im CREATE TABLE ist egal) 2 Punkte. Grau: +2 Punkte

```
CREATE TABLE KOPIE (
      id    NUMERIC(7)  NOT NULL,
      einkaufspreis     NUMERIC(5,2)      NOT NULL,
      standort    VARCHAR(50) NOT NULL,
      defekt      VARCHAR(100),
      anzahldvds  NUMERIC(2),
      CONSTRAINT  kopie_id_pk PRIMARY KEY(id)
);

CREATE TABLE TITEL (
      id    NUMERIC(7)  NOT NULL,
      titel VARCHAR(150)      NOT NULL,
      datum DATE  NOT NULL,
      fskfreigabe NUMERIC(2)  NOT NULL,
      kritik      LONGTEXT,
      tit_typ     VARCHAR(5),
      fil_regisseur     VARCHAR(50),
      fil_extras  VARCHAR(250),
      fil_genre   VARCHAR(50),
      gam_firma   VARCHAR(50),
      gam_plattform     VARCHAR(30),
      gam_kategorie     VARCHAR(50),
      preisstufe_id     NUMERIC(3)  NOT NULL,
      CONSTRAINT  titel_id_pk PRIMARY KEY(id)
);

ALTER TABLE TITEL
      ADD CONSTRAINT    titel_preisstufe_fk
      FOREIGN KEY (preisstufe_id)
      REFERENCES  PREISSTUFE(id);


CREATE TABLE TITEL_KOPIE (
      titel_id    NUMERIC(7)  NOT NULL,
      kopie_id    NUMERIC(7)  NOT NULL,
      CONSTRAINT  titel_kopie_id_pk PRIMARY KEY(titel_id, kopie_id)
);

ALTER TABLE TITEL_KOPIE
      ADD CONSTRAINT    titel_kopie_titel_fk
      FOREIGN KEY (titel_id)
      REFERENCES  TITEL(id);
```



```
ALTER TABLE TITEL_KOPIE
      ADD CONSTRAINT    titel_kopie_kopie_fk
      FOREIGN KEY (kopie_id)
      REFERENCES  KOPIE(id);

CREATE TABLE PREISSTUFE (
      id    NUMERIC(3)  NOT NULL,
      gebuehr     NUMERIC(5,2)      NOT NULL,
      notiz VARCHAR(100),
      CONSTRAINT  preisstufe_id_pk  PRIMARY KEY(id)
);

CREATE TABLE VORMERKUNG (
      id    NUMERIC(8)  NOT NULL,
      abgelaufen  DATE,
      benachrichtigt    DATE,
      kopie_id    NUMERIC(7),
      titel_id    NUMERIC(7)  NOT NULL,
      CONSTRAINT  vormerkung_id_pk  PRIMARY KEY(id)
);

ALTER TABLE VORMERKUNG
      ADD CONSTRAINT    vormerkung_kopie_fk
      FOREIGN KEY (kopie_id)
      REFERENCES  KOPIE(id);

ALTER TABLE VORMERKUNG
      ADD CONSTRAINT    vormerkung_titel_fk
      FOREIGN KEY (titel_id)
      REFERENCES  TITEL(id);
```

Aufgabe 4: Queries **(19 Punkte)**

a) 2 Punkte

```
SELECT VERTRETER.v_id, VERTRETER.v_name, VERTRETER.v_vorname
FROM VERTRETER
WHERE VERTRETER.v_werber IS NOT NULL
```

b) 3 Punkte

```
SELECT
     INSTRUMENT.i_name AS Umsatzverursacher,
     SUM(INSTRUMENT.i_preis*UMSATZ.i_anz) AS Umsatz
FROM INSTRUMENT NATURAL JOIN UMSATZ
WHERE INSTRUMENT.i_name = 'Mikrofon'
GROUP BY Umsatzverursacher
```

c) 4 Punkte

```
SELECT
     WERBER.v_id          AS Werber_id,
     WERBER.v_name        AS Werber_Name,
     WERBER.v_vorname     AS Werber_Vorname,
     GEWORBENER.v_id      AS Geworbener_id,
     GEWORBENER.v_name    AS Geworbener_Name,
     GEWORBENER.v_vorname      AS Geworbener_Vorname
FROM VERTRETER AS GEWORBENER RIGHT OUTER JOIN VERTRETER AS
WERBER
ON GEWORBENER.v_werber = WERBER.v_id
```

d) 5 Punkte

```
SELECT
     VERTRETER.v_id AS 'ID Nummer',
     VERTRETER.v_name AS 'Name',
     VERTRETER.v_vorname AS 'Vorname'
FROM VERTRETER
WHERE VERTRETER.v_id =
     (SELECT VERTRETER.v_werber
      FROM VERTRETER
      WHERE VERTRETER.v_id =
          (SELECT UMSATZ.v_id
           FROM UMSATZ
           WHERE UMSATZ.i_anz = 10
           AND UMSATZ.datum = '2015-06_24'
           AND UMSATZ.i_id IN
               (SELECT INSTRUMENT.i_id
                FROM INSTRUMENT
                WHERE INSTRUMENT.i_name = 'Gibson Les Paul'
               )
          )
     )
```


e) 5 Punkte

```
SELECT v_name, v_vorname, v_anschrift
FROM VERTRETER NATURAL JOIN UMSATZ NATURAL JOIN INSTRUMENT
WHERE     UMSATZ.datum = '2015-05-26'
AND       INSTRUMENT.i_name = 'Warwick Triumph'
OR        INSTRUMENT.i_name = 'Gibson Les Paul'
;
```

INSTRUMENT [i_name = 'Warwick Triumph' V i_name = 'Gibson Les Paul'] NATJOIN UMSATZ [datum = '2015-05-26'] NATJOIN VERTRETER [v_name, v_vorname, v_anschrift]

Aufgabe 5: Datenmanipulation und Schemamanipulation (10 Punkte)

a) 1 Punkt

```
INSERT INTO VERTRETER (id, v_name, v_vorname, v_werber,
v_anschrift)
VALUES (3456, 'Baumann', 'Paul', 1234, 'Universitätsstraße 1,
58097 Hagen')
```

b) 3 Punkte

```
UPDATE INSTRUMENT
SET INSTRUMENT.i_name = 'Neumann U87'
WHERE INSTRUMENT.i_name = 'Mikrofon'
AND INSTRUMENT.i_preis =
 (SELECT MAX(INSTRUMENT.i_preis)
  FROM INSTRUMENT
  WHERE INSTRUMENT.i_name = 'Mikrofon')
```

c) 2 Punkte

```
ALTER TABLE VERTRETER
     ADD COLUMN v_btlg NUMERIC (3,2)
```

d) 4 Punkte

Konto (k_nummer, k_blz, k_art, v_id)

```
CREATE TABLE KONTO (
      k_nummer    NUMERIC(15) ,
      k_blz NUMERIC(8)  NOT NULL,
      k_art VARCHAR(15),
      CONSTRAINT  konto_k_nummer_pk PRIMARY KEY(k_nummer)
);

ALTER TABLE KONTO
      ADD CONSTRAINT    konto_vertreter_fk
      FOREIGN KEY (v_id)
      REFERENCES  VERTRETER(v_id);
```

Hinweis: Das Foreign Key Constraint kann natürlich auch in das `CREATE TABLE` Statement integriert werden.

Aufgabe 6: Funktionale Abhängigkeiten, Normalformen (10 Punkte)

Schlüssel der Relation Produkt ist Produkt-Nr, Hersteller-Nr. Es gibt aber keine Attribute, die voll funktional von diesem Schlüssel abhängen. Daher wird diese Relation in 3 Relationen aufgeteilt:

Produkt:
Produkt-Nr, P-Bezeichung, P-Name

Hersteller:
Hersteller-Nr, H-Name

Lieferung:
Produkt-Nr, Hersteller-Nr, Lieferzeit

Diese Relationen sind in 3NF, da keine transitiven Abhängigkeiten vorliegen.

Schlüssel der Relation Lieferant ist Produkt-Nr, Lieferanten-Nr. Es gibt auch hier Attribute, die nur von Lieferanten-Nr abhängig sind und damit nicht vollständig abhängig sind vom Schlüssel. Daher wird diese Relation in 2 Relationen aufgeteilt:

Lieferant:
Lieferanten-Nr, L-Name, L-Adresse, L-Zertifikat-Typ, L-Zertifikat-Beschreibung

Preis:
Produkt-Nr, Lieferanten-Nr, Preis

Die neue Relation Lieferant ist in 2NF, da es noch transitive Abhängigkeiten gibt:
Lieferanten-Nr → L-Zertifikat-Typ → L-Zertifikat-Beschreibung

Daher wird diese Relation ein weiteres Mal aufgespalten:

Lieferant:
Lieferanten-Nr, L-Name, L-Adresse

Lieferanten-Zertifikate
L-Zertifikat-Typ, L-Zertifikat-Beschreibung

Aufgabe 7: 3-Schichten-Architektur eines DBS **(10 Punkte)**

Die 3 Datenebenen heißen: (1 Punkt)
- externes Modell
- konzeptuelles Modell
- internes Modell

a) Das externe Modell: (2 Punkte)
Das externe Modell beschreibt die einzelnen Sichten der Benutzer. Jeder Benutzer soll nur die für ihn relevanten Daten sehen. Die Objekttypen und Beziehungstypen müssen nicht mit denen des konzeptuellen Modells identisch sein, sie müssen lediglich „inhaltlich" im konzeptuellen Modell enthalten sein. (1 Punkt)
Die Erstellung des entsprechenden Schemas hängt von den Anforderungen und auch von den Befugnissen (Zugriffsrechten usw.) des jeweiligen Benutzers ab. (1 Punkt)

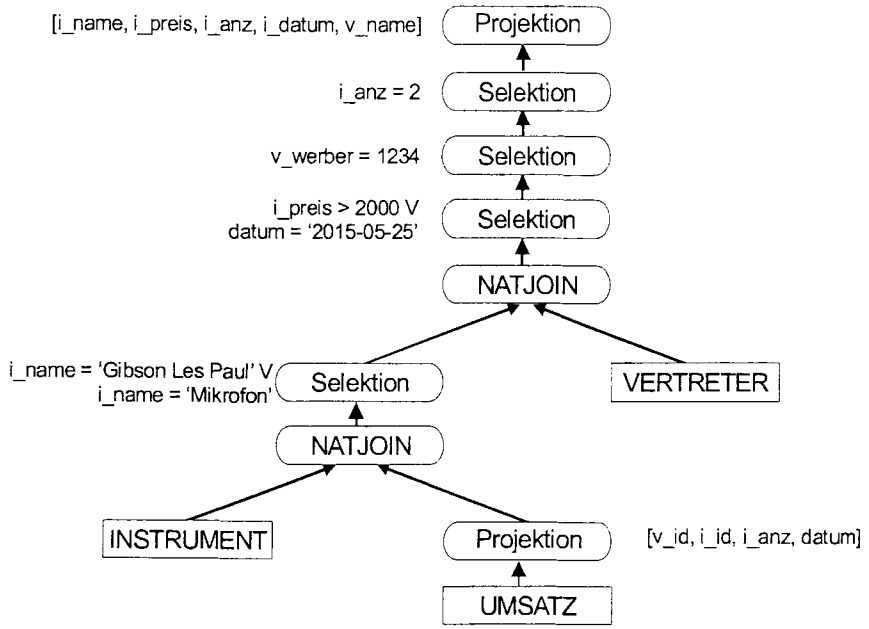
b) Das konzeptuelle Modell: (2 Punkte)
Das konzeptuelle Modell beschreibt die Gesamtheit aller Daten, die innerhalb der Datenbank verwaltet werden. Dazu gehören die Festlegungen von Beziehungen, Integritätsbedingungen und Operationen (1 Punkt).
Das zu erstellende Schema hängt einzig und allein von den in der Realität existierenden Strukturen (die wesentlichen Daten eines Unternehmens und die wesentlichen Beziehungen zwischen den Daten) ab (1 Punkt).

c) Das interne Modell: (3 Punkte)
Das interne Modell beschreibt die vom Datenbankadministrator festgelegte physische Datenorganisation. Es enthält u.a. alle Informationen über den Aufbau der abgespeicherten Daten, die Speicherung der Daten und über die Zugriffspfade (1 Punkt). Es muß genaue Festlegungen der folgenden Punkte enthalten:
- Repräsentation von Attributwerten
- Aufbau gespeicherter Sätze
- Zugriffsmethoden auf Sätze
- zusätzliche Zugriffspfade (Indizes, Verkettungen, usw.)
- Die Erstellung des internen Schemas hängt von den statischen Informationen über die Häufigkeit von Anwendungen, von Zugriffen auf Objekten, über Zeitbeschränkungen für Anwendungen, Verteilung von Werten für Attribute usw. ab. (2 Punkte)

Datenunabhängigkeit: (2 Punkte)
Datenunabhängigkeit bedeutet, daß Anwendungsprogramme von Änderungen auf der internen und der konzeptuellen Ebene unberührt bleiben. Man unterscheidet physische und logische Datenunabhängigkeit.
Physische Datenunabhängigkeit bedeutet Isolierung der Anwendungsprogramme vor Änderungen der physischen Datenorganisation. (1 Punkt)
Logische Datenunabhängigkeit bedeutet Isolierung der Anwendungsprogramme vor Änderungen des konzeptuellen Modells. (1 Punkt)

Aufgabe 8: Musterlösung **(13 Punkte)**

a) (3 Punkte)



b) Die Selektion [i_anz = 2] kann vor den JOINS ausgeführt werden, weil sie sich nur auf die Relation UMSATZ bezieht. Die Selektion, die sich nur auf die Relation INSTRUMENT bezieht, [i_name = 'Gibson Les Paul' V i_name = 'Mikrofon'] kann vor dem JOIN mit UMSATZ ausgeführt werden.
Die Selektion [v_werber = 1234] bezieht sich nur auf die Relation VERTRETER und kann vor dem JOIN mit dem Verbund aus UMSATZ und INSTRUMENT ausgeführt werden.
[i_preis > 2000 V datum > '2015-05-25'] kann wegen der 'oder' Verknüpfung nicht aufgespalten werden, bezieht sich aber nur auf UMSATZ und INSTRUMENT, kann also vor dem JOIN mit VERTRETER ausgeführt werden. (4 Punkte)

Optimierter Ausdruck: (3 Punkte)

INSTRUMENT [i_name = 'Gibson Les Paul' V i_name = 'Mikrofon'] NATJOIN (UMSATZ [i_anz = 2] [v_id, i_id, i_anz, datum]) [i_preis > 2000 V datum > '2015-05-25'] NATJOIN (VERTRETER [v_werber = 1234]) [i_name, i_preis, i_anz, i_datum, v_name]

Operatorbaum des optimierten Ausdrucks (3 Punkte):