

1

Ganzheitliche Aufgabe I Fachqualifikationen

Allgemeine Korrekturhinweise

Die Lösungs- und Bewertungshinweise zu den einzelnen Handlungsschritten sind als Korrekturhilfen zu verstehen und erheben nicht in jedem Fall Anspruch auf Vollständigkeit und Ausschließlichkeit. Neben hier beispielhaft angeführten Lösungsmöglichkeiten sind auch andere sach- und fachgerechte Lösungsalternativen bzw. Darstellungsformen mit der vorgesehenen Punktzahl zu bewerten. Der Bewertungsspielraum des Korrektors (z. B. hinsichtlich der Berücksichtigung regionaler oder branchenspezifischer Gegebenheiten) bleibt unberührt.

Zu beachten ist die unterschiedliche Dimension der Aufgabenstellung (nennen – erklären – beschreiben – erläutern usw.). Wird eine bestimmte Anzahl verlangt (z. B. „Nennen Sie fünf Merkmale ...“), so ist bei Aufzählung von fünf richtigen Merkmalen die volle vorgesehene Punktzahl zu geben, auch wenn im Lösungshinweis mehr als fünf Merkmale genannt sind. Bei Angabe von Teilpunkten in den Lösungshinweisen sind diese auch für richtig erbrachte Teilleistungen zu geben.

In den Fällen, in denen vom Prüfungsteilnehmer

- keiner der fünf Handlungsschritte ausdrücklich als „nicht bearbeitet“ gekennzeichnet wurde,
- der 5. Handlungsschritt bearbeitet wurde,
- einer der Handlungsschritte 1 bis 4 deutlich erkennbar nicht bearbeitet wurde,

ist der tatsächlich nicht bearbeitete Handlungsschritt von der Bewertung auszuschließen.

Ein weiterer Punktabzug für den bearbeiteten 5. Handlungsschritt soll in diesen Fällen allein wegen des Verstoßes gegen die Formvorschrift nicht erfolgen!

Für die Bewertung gilt folgender Punkte-Noten-Schlüssel:

Note 1 =	100 – 92 Punkte	Note 2 =	unter	92 – 81 Punkte
Note 3 =	unter 81 – 67 Punkte	Note 4 =	unter	67 – 50 Punkte
Note 5 =	unter 50 – 30 Punkte	Note 6 =	unter	30 – 0 Punkte

1. Handlungsschritt (25 Punkte)

aa) 2 Punkte

z. B.

Interview, Beobachtung, Fragebogen

ab) 4 Punkte

z. B.

- Kompatibel zum Betriebssystem
- Der geforderte Funktionsumfang muss gegeben sein.
- Datensicherheit/Datensicherung muss gewährleistet sein.
- Schnittstellen zu den anderen im Betrieb verwendeten Softwaresystemen müssen vorhanden sein.

ba) 8 Punkte

Sachebene	Beziehungsebene
Klärung der Projektziele mit den Beteiligten und Beseitigung von Missverständnissen	Gegenseitiges Kennenlernen der Projektmitarbeiter
Ausarbeitung eines groben Projektstrukturplans (PSP) und Projektterminplans (PTP)	Erkennen von Rollen im Projekt
Verteilung wichtiger Aufgaben	Entwickeln eines WIR-Gefühls
Festlegung der Projektorganisation sowie des Informations- und Kommunikationsweges	Aufdeckung von Konfliktpotenzial
Entwicklung erster Lösungsansätze	u. a.
Abschätzung von Projektrisiken	
u. a.	

bb) 3 Punkte

z. B.

- Projektbezogenes Entscheidungsrecht
- Projektbezogenes Weisungsrecht
- Mitwirkung bei der Zielfestlegung des Projektes
- Mitsprache bei der Auswahl von hinzuzuziehenden Fachpersonal aus den Abteilungen und externen Firmen
- Einberufung und Leitung von Projektsitzungen
- Akzeptieren oder zurückweisen von Projektergebnissen

ca) 6 Punkte

Kriterium	Gewichtung (G)	Anbieter A		Anbieter B	
		Erfüllung (E)	Nutzwert (N)	Erfüllung (E)	Nutzwert (N)
Image des Softwareanbieters	25	1	25	3	75
Service des Softwareanbieters	30	1	30	2	60
Kompatibilität der Software	10	3	30	1	10
Ergonomie der Software	10	1	10	3	30
Customizing der Software	5	3	15	1	5
Beratungskompetenz des Anbieters	20	3	60	1	20
S U M M E	100		170		200

Anbieter B würde den Zuschlag erhalten.

cb) 2 Punkte

z. B.

Subjektivität beeinflusst die Bewertung

2. Handlungsschritt (25 Punkte)

erzeugeListe(persnr: int, zeiten: zweidim Tabelle vom Typ int, jahr : int, monat : int)

```
// Variablen
anwesenheitTag
anwesenheitMonat := 0
tag := 1
zeile := 0

schreibeKopf (persnr, jahr, monat)

solange (zeile < Anzahl Zeilen in zeiten)
  anwesenheitTag := 0
  // Keine Buchung
  wenn tag < zeiten[zeile][0]
    dann
      // Keine Buchung
      schreibeZeile(tag, -1, -1, -1, -1, anwesenheitTag, " nicht anwesend")
    sonst
      // Zwei Buchungen
      wenn tag = zeiten[zeile][0] und tag = zeiten[zeile + 1][0]
        dann
          anwesenheitTag := zeiten[zeile+1][1] * 60 + zeiten[zeile+1][2]
                           - zeiten[zeile][1] * 60 - zeiten[zeile][2]
          anwesenheitMonat := anwesenheitMonat + anwesenheitTag;
          schreibeZeile(tag, zeiten[zeile][1], zeiten[zeile][2],
                        zeiten[zeile+1][1], zeiten[zeile+1][2],
                        anwesenheitTag,
                        "")
          zeile := zeile + 2
        sonst
          // Eine Buchung
          schreibeZeile(tag,
                        zeiten[zeile][1], zeiten[zeile][2],
                        -1, -1
                        anwesenheitTag,
                        " Buchung fehlt");
          zeile := zeile + 1
        endewenn
      endewenn
    tag := tag + 1;
  ende solange

// Keine Buchungen am Monatsende, Tageszähler ist kleiner als die Anzahl der
// abgelaufenen Tage im Monat

solange tag <= tageImMonat(monat,jahr)
  schreibeZeile(tag, -1, -1, -1, -1, anwesenheitTag, "nicht anwesend")
  tag := tag + 1
ende solange

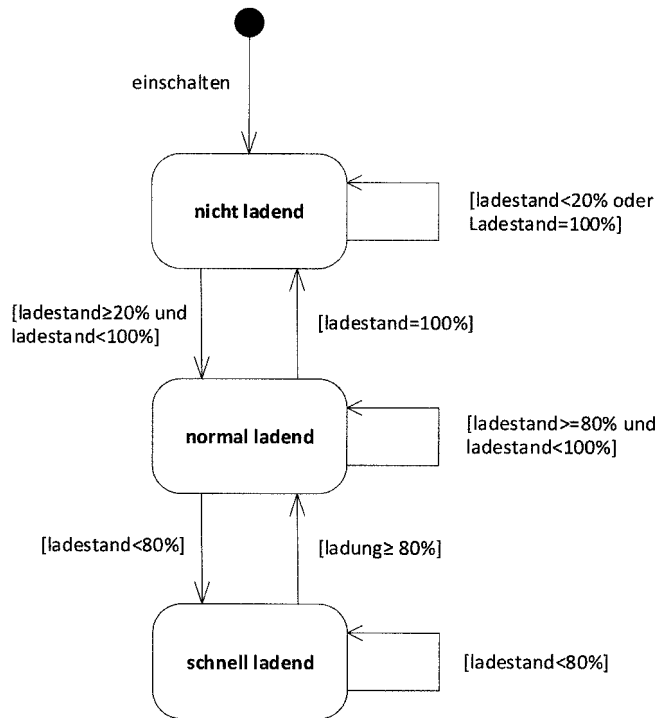
// Fusszeile ausgeben
schreibeFusszeile(anwesenheitMonat);
```

Andere Lösungen sind möglich.

3. Handlungsschritt (25 Punkte)

a) 16 Punkte

4 x je 1 Punkt pro Zustand (inklusive Startzustand), 8 x je 1,5 Punkte pro Transition mit Bedingung



ba) 2 Punkte

1 Punkt für „setZustand“ oder „zustand:=“, 1 Punkt Aufruf des Objektgetters

+ Ladegerät()

entweder über den Setter:

setZustand(NichtLadend.getNichtLadend())

oder direkt:

zustand:= NichtLadend.getNichtLadend()

bb) 3 Punkte

1 Punkt Bedingung, 2 Punkte Methodenaufrufe

+ bearbeiten(ladegerät : Ladegerät) : void

wenn (ladegerät.getLadestand() >= 20 und ladegerät.getLadestand() < 100)

dann ladegerät.setZustand(NormalLadend.getNormalLadend())

bc) 4 Punkte

Die Klasse Ladegerät hält die polymorphe Referenz *zustand* auf die abstrakte Klasse *Zustand* (1 Punkt). Dieser Referenz werden zur Laufzeit konkrete Objekte der Klassen *NichtLadend*, *NormalLadend* bzw. *SchnellLadend* zugewiesen (1 Punkt). Das ist möglich, da die konkreten Zustandsklassen von der abstrakten Klasse *Zustand* abgeleitet sind (1 Punkt). Je nachdem, welchen konkreten Zustand *zustand* gerade referenziert, wird mit *zustand.bearbeiten(this)* immer das entsprechende zustandsspezifische Verhalten ausgeführt (spätes/dynamisches Binden) (1 Punkt).

4. Handlungsschritt (25 Punkte)

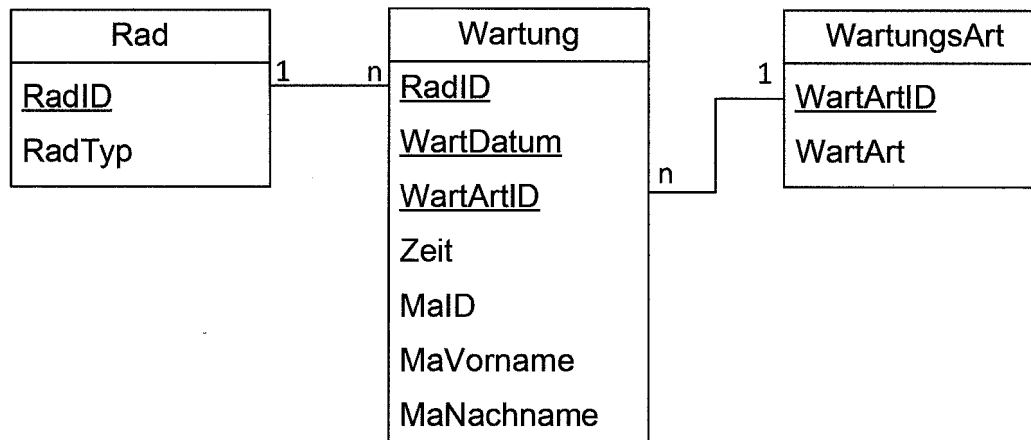
a) 9 Punkte

3 Punkte für Auflösung des Mitarbeiternamens, 3 Punkte für Auflösung der Wiederholgruppen, 3 Punkte für zusammengesetzten Primärschlüssel (RadID, WartDat, WartArtID).

<u>RadID</u>	RadTyp	<u>WartDat</u>	<u>WartArtID</u>	WartArt	Zeit	MaID	MaVorname	MaNachname
E5	E-Bike 400	2019-10-17	12	Bremse	30	123	Klaus	Müller
E5	E-Bike 400	2019-10-17	09	Schaltung	12	345	Beatrice	Richter
E5	E-Bike 400	2019-10-17	05	Akku	15	456	Kurt	Helmig
C2	Citybike 28	2019-10-20	03	Lager	25	345	Beatrice	Richter
C2	Citybike 28	2019-10-20	12	Bremse	10	123	Klaus	Müller
E5	E-Bike 400	2019-11-15	09	Schaltung	15	123	Klaus	Müller

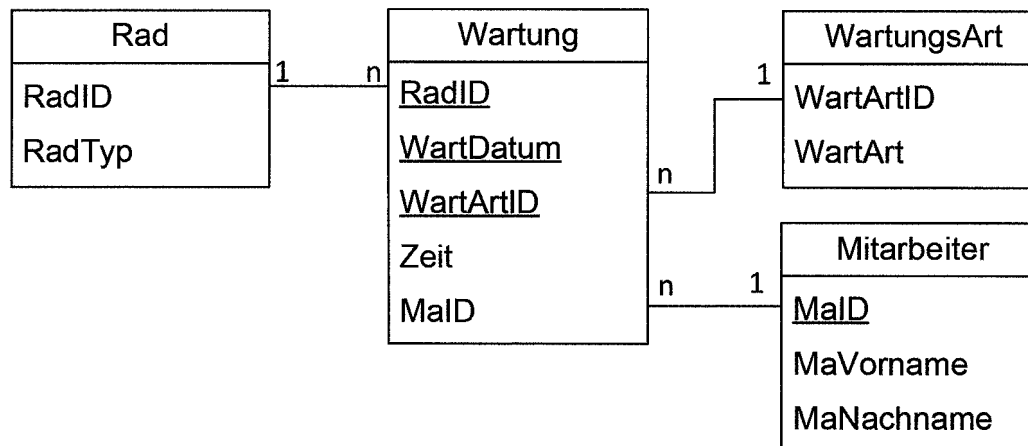
b) 11 Punkte

Je 1 Punkt pro Tabellennamen, je 1 Punkt für Beziehung, 1 Punkt für Kombischlüssel in Wartung, je 1 Punkt für Schlüssel in Rad und WartungsArt, je 0,5 Punkte pro Attribut



c) 5 Punkte

1 Punkt für Tabellennamen Mitarbeiter, je 1 Punkt für Schlüssel MaID in Mitarbeiter und MaID in Rad_Wartung, 1 Punkt für Beziehung, je 0,5 Punkte für Attribute in Mitarbeiter



Hinweis: Bei richtigem Datenbankmodell, aber falscher Zuordnung zur zweiten und dritten Normalform pauschal 3 Punkte abziehen.

5. Handlungsschritt (25 Punkte)

aa) 2 Punkte

```
CREATE TABLE Defekt (  
    DefektID int,  
    Beschreibung VARCHAR(25),  
    PRIMARY KEY (DefektID)  
);
```

ab) 3 Punkte

```
CREATE TABLE DefektBuchung (  
    DefektID int,  
    KdID int,  
    VRadID int,  
    Datum Date,  
    FOREIGN KEY (DefektID) REFERENCES Defekt(DefektID),  
    FOREIGN KEY (KdID, VRadID, Datum)  
        REFERENCES Buchung(KdID, VRadID, Datum),  
    PRIMARY KEY (KdID, VRadID, Datum, DefektID));
```

b) 5 Punkte

```
SELECT r.RadTypID, COUNT(r.RadTypID) Anzahl  
FROM RadTyp r, VerleihRad v, Buchung b  
WHERE r.RadTypID = v.RadTypID AND v.VRadID = b.VRadID  
GROUP BY r.RadTypID HAVING Anzahl > 10;
```

c) 5 Punkte

```
SELECT k.KdID, SUM(b.Tage * r.RadTypPreis) Umsatz  
FROM Kunde k, Buchung b, VerleihRad v, RadTyp r  
WHERE k.KdID = b.KdID AND b.VRadID = v.VRadID AND v.RadTypID = r.RadTypID  
GROUP BY k.KdID ORDER BY Umsatz DESC;
```

d) 5 Punkte

```
SELECT r.RadTypID, r.RadTypBez, r.RadTypPreis  
FROM RadTyp r  
WHERE r.RadTypPreis > (  
    SELECT RadTypPreis FROM RadTyp WHERE RadTypID = 1001);
```

e) 5 Punkte

```
SELECT MONTH(datum) Monat,  
    100 * COUNT(*) /  
        (SELECT COUNT(*) FROM Buchung WHERE YEAR(datum) = 2019) Anteil  
FROM Buchung  
WHERE YEAR(datum) = 2019  
GROUP BY Monat;
```