Projektbericht: Ersatzteilmanagement

Datenbank-Programmierung 2020

Jonathan Seifert

1. August 2020

Link zu git-Repository
`https://gitlab-bs.kube.informatik.uni-halle.de/ajtrs/dbp20-projekt`

Inhaltsverzeichnis

1 Vorstellung der Anwendung

1.1 Anwendungsdomäne

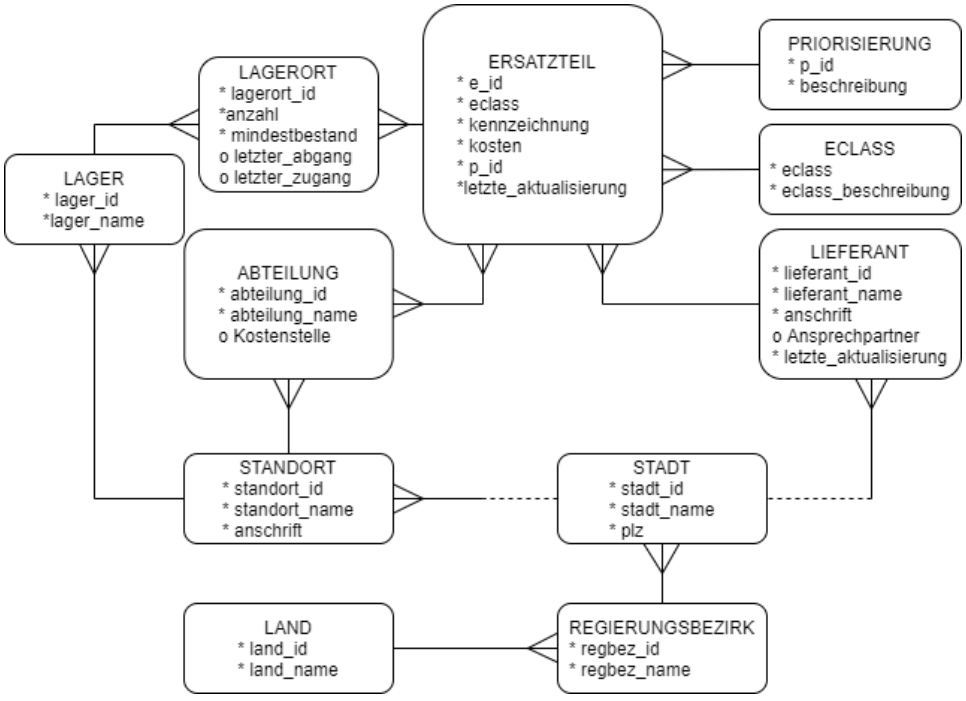
Ersatzteile können aufgrund ihrer Wichtigkeit sowie Lieferzeiten in Kategorien eingeteilt werden um die Lagerhaltung zu optimieren und somit unnötige Kosten zu vermeiden. Ausserdem versichert die Haltung der Daten in Datenbanken, dass jederzeit auf sie zugegriffen werden können.
Die Daten in einer Datenbank zu speichern hat den Vorteil, dass so eine schnelle Auswertung sowie Aktualisierung der Ersatzteildaten automatisiert erfolgen kann.

1.2 Fiktive Anwendungsprogramme

Ein fiktives Anwendungsprogramm kann die Lagersituation eines Ersatzteils an einem Lagerort überwachen. Lagermitarbeiter können die Anzahl der gelagerten Teile verändern und somit die Datenbank aktuell halten.
Eine andere Anwendungsmöglichkeit ist die DataWarehouse-Sicht auf die Ersatzteile und Lagerorte. Reports in Bezug auf die Herkunft und Kosten der Ersatzteile(Sortierung nach Lieferant), Art der gelagerten Teile(eclass-Kategorisierung), den Lagerorten und Abteilungszuordnungen werden durch dieses Datenbanksystem unterstützt.

2 Datenbank

2.1 Schema



2.1.1 Tabellen in Kurznotation mit Erklärung

1. *ersatzteil*(e_id, eclass →eclass, lieferant_id →lieferant, kennzeichnung, kosten, p_id →priorisierung, letzte_aktualisierung)

   Verwaltung der Ersatzteile und Zuordnung zu Lieferanten sowie Klassisfikation nach Liefersituation/Wichtigkeit

2. *lagerort*(lagerort_id, e_id →ersatzteil, lager_id →lager, anzahl, mindestbestand, letzter_abgang^0, letzter _zugang^0)

   Folge der $n : m$-Beziehung zwischen *lager*(...) und *ersatzteil*(...)

   Verwaltung der Ersatzteile in verschiedenen Lagern mit aktueller Anzahl, Mindestbestand sowie Zeitpunkt des letzten Abgangs/Zugangs

3. *lieferant*(lieferant_id, lieferant_name, anschrift, stadt_id →stadt, ansprechpartner^0, letzte_aktualisierung)

   Verwaltung der Ersatzteillieferanten

4. *lager*(lager_id, standort_id →standort, lager_name)

   Verwaltung der Lager, in denen die Ersatzteile gelagert werden.

5. *abteilung*(abteilung_id, standort_id →standort, abteilung_name)

   Verwaltung der Abteilungen in den Standorten.

6. *zuordnung*(e_id →ersatzteil, abteilung_id →abteilung)

   Folge der $n : m$-Beziehung zwischen *ersatzteil*(...) und *abteilung*(...)

   Verwaltung der Zuordnung der Ersatzteile zu den Abetilungen

7. *standort*(standort_id, standort _name, stadt_id →stadt, anschrift)

   Verwaltung der Standorte inklusive Name und Anschrift.

8. *stadt*(stadt_id, regbez_id →regierungsbezirk, stadt_name, plz)

   Verwaltung der Städte der verschiedenen Regierungsbezirke.

9. *regierungsbezirk* (regbez_id, land_id →land, regbez _name)

   Verwaltung der Regierungsbezirke der einzelnen Länder.

10. *land*(land_id, land_name)

    Verwaltung der Länder.

11. *eclass*(eclass, eclass_beschreibung)

    Erklärung
    `https://www.eclass.eu/standard/eclss-kurz-erklaert.html`

Liste
`https://www.eclasscontent.com/index.php`

12. *priorisierung*(p_id, beschreibung)

    Ersatzteile können nach Wichtigkeit/Lieferzeit priorisiert werden

    a Ersatzteil muss auf Lager sein

    b Lieferzeit zwischen einem Tag und einer Woche

    c Lieferzeit von mehr als einer Woche

2.1.2 Tabelle für Ersatzteile

```
CREATE TABLE ersatzteil
  e_id NUMERIC(5) PRIMARY KEY DEFAULT NEXTVAL('e_id_seq') ON DELETE CASCADE,
  eclass CARCHAR(11) NOT NULL REFERENCES eclass,
  lieferant_id NUMERIC(3) NOT NULL REFERENCES lieferant,
  kennzeichnung VARCHAR(50) NOT NULL,
  kosten NUMERIC(9,2) NOT NULL,
  p_id CHARACTER(1) NOT NULL REFERENECES priorisierung,
  letzte_aktualisierung TIMESTAMP(0) WITHOUT TIME ZONE NOT NULL DEFAULT now(),
  UNIQUE(eclass, kennzeichnung),
  CONSTRAINT kosten_muessen_positiv_sein CHECK (kosten > 0)
);
CRATE SEQUENCE e_id_seq AS INTEGER START 1 INCREMENT 1 MAXVALUE 99999;

CREATE FUNCTION aktualisiert() RETURNS TRIGGER
LANGUAGE plpgsql
AS $$
  BEGIN
  NEW.letzte_aktualisierung = CURRENT_TIMESTAMP;
  RETURN NEW
  END
   $$;
CREATE TRIGGER aktualisierung_ersatzteil BEFORE UPDATE ON ersatzteil
FOR EACH ROW EXECUTE aktualisiert();
```

- **E_Id**
  - wird automatisch per Sequenz generiert
  - Annahme: nicht mehr als 99999 verschiedene Ersatzteile

  → *NUMERIC(5)*

  - Sequenz e_id_seq: INTEGER (99999 > 32767) Maxvalue 99999 (Höchster Wert von Numeric(5))

  - löscht Einträge in der Lagerort-Tabelle, sobald das Ersatzteil oder der zugehörige Lieferant gelöscht wird

- **eclass** →Tabelle für eclass-Kategorisierung
- **Lieferant_Id** →Tabelle für Lieferanten
- **Kennzeichnung**
  - Annahme: Kennzeichnung ncht länger als 50 Zeichen

  → *VARCHAR(50*
- **Kosten**
  - Annahme: Kosten pro Ersatzteil übersteigen 9999999,99 nicht
  - *NUMERIC(9,2)*
  - Constraint Kosten_muessen_positiv_sein versichtert, dass Kosten mind. 0,01 betragen
- **P_Id** →Tabelle für Priorisierung der Lagersituation
- **Letzte_Aktualisierung**
  - Spaltenwert wird beim Einfügen und Verändern der Spalte auf den aktuellen Zeitpunkt gesetzt (geschieht automatisch)

2.1.3 Tabelle für Lagerorte (*n:m*-Beziehung)

```
CREATE TABLE lagerort(
  lagerort_id NUMERIC(7) PRIMARY KEY,
  e_id NUMERIC(5) NOT NULL REFERENCES ersatzteil,
  lager_id VARCHAR(4) NOT NULL REFERENCES lager,
  anzahl NUMERIC(2) NOT NULL,
  mindestbestand NUMERIC(2) NOT NULL,
  letzter_abgang TIMESTAMP(0) WITHOUT TIME ZONE,
  letzter_zugang TIMESTAMO(0) WITHOUT TIME ZONE,
  CONSTRAINT anzahl_darf_nicht_negativ_sein CHECK (anzahl>-1),
  CONSTRAINT mindestbestand_darf_nicht_negativ_sein
  CONSTRAINT (mindestbestand>-1),
  UNIQUE(lager_id, e_id)
);
CREATE SEQUENCE lagerort_id_seq AS INTEGER
START 1 INCREMENT 1 MAXVALUE 9999999;

CREATE TRIGGER entnahme BEFORE UPDATE OF anzahl ON lagerort
FOR EACH ROW EXECUTE PROCEDURE teil_entnommen();
```

```
CREATE TRIGGER zufuehrung BEFORE UPDATE OF anzahl ON lagerort
FOR EACH ROW EXECUTE PROCEDURE teil_zugefuehrt();
```

- **Lagerort_Id**
  - Annahme: Maximal 99999 verchiedene Ersatzteile, diese können in 99 verschiedenen Lagern gerlagert werden (Worst-Case Szenario)

  → *NUMERIC(7)*
- **E_Id** →Tabelle für Ersatzteile
- **Lager_Id** →Tabelle für Lager
- **Anzahl**
  - Annahme: nicht mehr als 99 Ersatzteile des gleichen Typs in einem Lager

  → *NUMERIC(2)*
- **Mindesbestand**
  - Mindesbestand für ein Ersatzteil x im lager y
  - Annahme: Mindestbestand übersteigt nie 99

  → *NUMERIC(2)*
- **Letzter_Abgang**
  - *teil_entnommen()* und *entnahme* aktualisieren automatisch 'letzter_abgang', sobald die Anzahl in der Zeile verringert wurde
- **Letzter_Zugang**
  - *teil_zugefuehrt()* und *zufuehrung* aktualisieren automatisch 'letzter_zugang', sobald die Anzahl in der Zeile erhöht wurde

2.1.4 Tabelle für Lieferanten

```
CREATE TABLE lieferant(
  lieferant_id NUMERIC(3) PRIMARY KEY DEFAULT (NEXTVAL('lieferant_id_seq'))
      ON DELETE CASCADE,
  lieferant_name VARCHAR(50) NOT NULL,
  stadt_id NUMERIC(3) NOT NULL REFERENCES stadt,
  anschrift VARCHAR(50) NOT NULL,
  email VARCHAR(50) NOT NULL,
  ansprechpartner VARCHAR(50),
  letzte_aktualisierung TIMESTAMP(0) WITHOUT TIME ZONE NOT NULL DEFAULT now(),
  CONSTRAINT email_format CHECK (email like %@%.__%)
);
CREATE SEQUENCE lieferant_id_seq AS SMALLINT START 1 INCREMENT 1 MAXVALUE 999;
```

```
CREATE TRIGGER aktualisierung_lieferant BEFORE UPDATE ON lieferant
FOR EACH ROW EXECUTE PROCEDURE aktualisiert();
```

- **Lieferant_Id**
  - wird automatisch per Sequenz erzeugt
  - Annahme: nicht mehr als 999 verschiedene Lieferanten benötigt

  → *NUMERIC(3)*
  - Sequenz lieferant_id_seq:
    Smallint (999 < 32767) Maxvalue 999 (höchster Wert von Numeric(3))
  - beim Löschen werden ebenfalls die zugehörigen Ersatzteile entfernt
- **Lieferant_Name**
  - Annahme: Lieferantenname nicht länger als 50 Zeichen

  → *VARCHAR(50)*
- **Stadt_Id** →Tabelle für Städte
- **Anschrift**
  - Annahme: Anschrift ist nicht länger als 50 Zeichen

  → *VARCHAR(50)*
- **email**
  - Annahme: email nicht länger als 50 Zeichen

  → *VARCHAR(50)*
  - Constraint email_format stellt sicher, dass eine email eine Endung sowie ein @-Zeichen besitzt
- **Letzte_Aktualisierung**
  - *aktualisierung_lieferant* und *aktualisiert()*(→Wiederverwendung der Funktion von Tabelle für Ersatzteile setzten den Wert der Spalte auf den aktuellen Zeitpunkt, sobald ein anderer Spaltenwert dieser Zeile verändert wurde

  → *TIMESTAMP(0) WITHOUT TIME ZONE*
  - *DEFAULT now()*→ garantiert, dass auch beim Zeileneinfügen der aktuelle Zeitpunkt eingesetzt wird

2.1.5 Tabelle für Lager

```
CREATE TABLE lager(
  lager_id CHARACTER(4) PRIMARY KEY,
  standort_id CHARACTER(2) NOT NULL REFERENCES standort,
  lager_name VARCHAR(50) NOT NULL
);
```

- **Lager_Id**
  - Annahme: nicht mehr als (99-Anzahl Abteilungen) Lager pro Standort
  - Kombination aus Standort_Id und 2-stelliger Zahlenfolge ⇒ 4 Stellen

  → *CHARACTER(4)*
- **Standort_Id** →Tabelle für Standorte
- **Lager_Name**
  - Annahme: Lagername nicht länger als 50 Zeichen

  → *VARCHAR(50)*

2.1.6 Tabelle für Abteilungen

```
CREATE TABLE abteilung(
  abteilung_id CHARACTER(4) PRIMARY KEY,
  standort_id CHARACTER(2) NOT NULL  REFERENCES standort,
  abteilung_name VARCHAR(50)
);
```

- **Abteilung_Id**
  - Annahme: nicht mehr als (99-Anzahl Lager) Abteilungen pro Standort
  - Kombination aus Standort_Id und 2-stelliger Zahlenfolge ⇒ 4 Stellen

  → *CHARACTER(4)*
- **Standort_Id** →Tabelle für Standorte
- **Abteilung_Name**
  - Annahme: Längster Abteilungsname max. 50 Zeichen lang
  - *VARCHAR(50)*

2.1.7 Tabelle für Zuordnung der Ersatzteile zu Abteilungen

```
CREATE TABLE zuordnung(
  e_id NUMERIC(5) NOT NULL,
  abteilung_id CHARACTER(4) NOT NULL,
  UNIQUE(e_id, abteilung_id)
);
```

- **E_Id** →Tabelle für Ersatzteile
- **Abteilung_Id** →Tabelle für Abteilungen
- **UNIQUE(e_id, abteilung_id**

→ ein Ersatzteil kann einer Abteilung nicht mehrmals zugeordnet werden

2.1.8 Tabelle für Standorte

```
CREATE TABLE standort(
  standort_id CHARACTER(2) PRIMARY KEY
  stadt_id NUMERIC(4)NOT NULL REFERENCES stadt,
  standort_name VARCHAR(50) NOT NULL,
  anschrift  VARCHAR(50)
```

- **Standort_Id**
  - Annahme: nicht mehr als 99 Standorte
  - 2-stellige Zahlenkombination, kann mit 0 beginnen

  → *CHARACTER(2)*
- **Stadt_Id** →Tabelle für Städte
- **Standort_Name**
  - Annahme: Standortname nicht länger als 50 Zeichen

  → *VARCHAR(50)*
- **Anschrift**
  - Annahme: Anschrift nicht länger als 50 Zeichen

  → *VARCHAR(50)*

2.1.9 Tabelle für Städte

```
CREATE TABLE stadt(
  stadt_id NUMERIC(4) PRIMARY KEY DEFAULT NEXTVAL('stadt_id_seq'),
  regbez_id VARCHAR(6) NOT NULL REFERENCES regierungsbezirk,
  stadt_name VARCHAR(58) NOT NULL,
  plz VARCHAR(10) NOT NULL
);
CREATE SEQUENCE stadt_id_seq AS SMALLINT START 1 INCREMET 1 MAXVALUE 9999;
```

- **Stadt_Id**
  - Annahme: nicht mehr als 9999 Städte
  - wird automatisch per Sequenz erzeugt
  - Sequenz stadt_id_seq: Smallint (9999 $<$ 32767), Maxvalue 9999 (höchster Wert für Numeric(4))
- **Regbez_Id** →Tabelle für Regierungsbezirke
- **Stadt_Name**

- längster Stadtname weltweit:
  Llanfairpwllgwyngyllgogerychwyrndrobwllllantysiliogogogoch
  (Wales, 58 Zeichen)

→ *Numeric(58)*

- **Plz**
  - USA hat längste Plz (besteht nicht nur aus Zahlen)

  → *VARCHAR(10)*

2.1.10 Tabelle für Regierungsbezirke

```
CREATE TABLE regierungsbezirk(
  regbez_id VARCHAR(6) PRIMARY KEY,
  land_id CHARACTER(2) NOT NULL REFERENCES land,
  regbez_name VARCHAR(50) NOT NULL,
  CONSTRAINT regierungsbezirk_format CHECK
   (regbez_id LIKE '___%'),
  UNIQUE(regbez_id, land_id)
);
```

- **Regbez_Id**
  - richtet sich nach dem ISO 3166 Standard
  - 3 Stellen für Land-Kennung(2) und Trennzeichen(1)
  - 1-3 Stellen für Regierungsbezirk-Kennung

  → *VARCHAR(6)*

- Land_Id →Tabelle für Länder
- **Regbez_Name**
  - Annahme: Name nicht länger als 50 Zeichen

  → *VARCHAR(50)*

- **UNIQUE(regbez_id, land_id**

→ jeden Regierungsbezirk gibt es nur ein mal im Land

2.1.11 Tabelle für Länder

```
CREATE TABLE land(
  land_id CHARACTER(2) PRIMARY KEY,
  land_name VARCHAR(50) NOT NULL,
  CONSTRAINT land_format CHECK
    (land_id SIMILAR TO '[A-Z][A-Z]'),
  UNIQUE(land_id, land)
);
```

- **Land_Id**
  - richtet sich nach dem ISO 3166-alpha2 Standard

  → *CHARACTER(2)*
- **Land_Name**
  - Annahme: Landname nicht länger als 50 Zeichen

  → *VARCHAR(50)*
- **UNIQUE(land_id, land_name)**

→ jedes Land_Id hat genau einen Namen ↔ jedes Land hat eine Land_Id

2.1.12 Tabelle für Priorisierung der Lagersituation

```
CREATE TABLE priorisierung
  p_id CHARACTER(1) PRIMARY KEY,
  beschreibung VARCHAR(45)
  CONSTRAINT a_b_c CHECK (p_id SIMILAR TO '[a-c]')
);
```

- **P_Id**
  - Kategorisierung der Lagersituation/Wichtigkeit in a, b und c
  - Constraint a_b_c stellt sicher, dass nur a, b und c als Primärschlüssel zugelassen werden, sodass keine Zeilen mehr hinzugefügt werden können

  → *CHARACTER(1)*
- **Beschreibung**
  - b hat mit 'Lieferzeit zwischen einer Woche und einem Monat' die längste Beschreibung

  → *VARCHAR(45)*

2.1.13 Tabelle für eclass-Kategorisierung

```
CREATE TABLE eclass(
  eclass VARCHAR(11) PRIMARY KEY,
  ecass_beschreibung VARCHAR(50) NOT NULL,
  CONSTRAINT eclass_format CHECK
   (eclass SIMILAR TO '[0-9][0-9]-[0-9][0-9]-[0-9][0-9]-[0-9][0-9]' OR
    eclass SIMILAR TO '[0-9][0-9]-[0-9][0-9]-[0-9][0-9]' OR
    eclass SIMILAR TO '[0-9][0-9]-[0-9][0-9]' OR
    eclass SIMILAR TO '[0-9][0-9]'),
    UNIQUE (eclass_beschreibung)
);
```

- **eclass**
  - `https://www.eclasscontent.com/index.php`
  - allgemeinstes Format: XX
  - speziellstes Format: XX-XX-XX-XX
  - X = 0,1,...,9

  → *VARCHAR(11)*
- **eclass_Beschreibung**
  - Annahme: eclass_beschreibung nicht länger als 50 Zeichen

  → *VARCHAR(50)*
- **UNIQUE(eclass_beschreibung)**

→ es gibt jede eclass-Beschreibung genau ein mal

2.1.14 Sequenzen

- *e_id_seq* →Tabelle für Ersatzteile
  (automatische Erzeugung der Ersatzteil-ID)
- *lieferant_id_seq* →Tabelle für Lieferanten
  (automatische Erzeugung der Lieferanten-ID)
- *lagerort_id_seq* →Tabelle für Lagerorte
  (automatische Erzeugung der Lagerort-ID)
- *stadt_id_seq* →Tabelle für Städte
  (automatische Erzeugung der Stadt-ID)

2.1.15 Trigger

- *aktualisierung_ersatzteil* →Tabelle für Ersatzteile
  führt Funktion *aktualisiert()* aus, sobald in der Tabelle ein Wert verändert wurde
- *aktualisierung_lieferant* →Tabelle für Lieferanten
  führt Funktion *aktualisiert()* aus, sobald in der Tabelle ein Wert verändert wurde
- *entnahme* →Tabelle für Lagerorte
  führt Funktion *teil_entnommen()* aus →aktualisiert den Wert in der Spalte *letzter_abgang* auf den Zeitpunkt, an dem der Wert 'anzahl' verringert wurde
- *zufuehrung* →Tabelle für Lagerorte
  führt Funktion *teil_zugefuehrt()* aus →aktualisiert den Wert in der Spalte *letzter_zugang* auf den Zeitpunkt, an dem der Wert 'anzahl' erhöht wurde

2.1.16 Nutzer

allgemeine Rechte

- mit der DB verbinden
- auf jedes Schema zugreifen

→ Rechteverteilung läuft über Schleife in */init/startup.sh*

```
GRANT CONNECT ON DATABASE etm TO <Nutzer>;
GRANT CONNECT ON SCHEMA zustand1 TO <Nutzer>;
GRANT CONNECT ON SCHEMA zustand2 TO <Nutzer>;
GRANT CONNECT ON SCHEMA zustand3 TO <Nutzer>;
```

Admin

```
CREATE USER admin WITH PASSWORD 'data';
ALTER USER admin WITH SUPERUSER;
```

- 'admin' hat SUPERUSER-Rechte →kann auf alle Postgres-Funktionen zugreifen

Lagerist

- darf mit SELECT auf alle Tabellen in allen Schemas zugreifen
- darf nicht auf die Kosten der Ersatzteile in der Tabelle <Schema>.ersatzteil zuzugreifen
- darf die Anzahl der vorhandenen Ersatzteile in der Tabelle <Schema>.lagerort zu aktualisieren

```
CREATE USER lagerist WITH PASSWORD 'logistik';
GRANT SELECT ON ALL TABLES IN SCHEMA {zustand1, zustand2, zustand3}
  TO lagerist;
REVOKE SELECT (kosten) ON TABLE {zustand1, zustand2, zustand3}.ersatzteil
  FROM lagerist;
GRANT UPDATE (anzahl) ON TABLE {zustand1, zustand2, zustand3}.lagerort
  TO lagerist;
```

Lagermitarbeiter sollen den Ersatzteilbestand in den Lagern, wenn benötigt, aktualisieren.

Abteilungsleiter

- darf mit SELECT auf alle Tabellen in allen Schemas zugreifen
- darf Zeilen hinzufügen, verändern und löschen in allen Schemas in den Tabellen
  - Lieferant
  - Ersatzteil
  - Lagerort
- hat alle Rechte bei den Sequenzen zur Erzeugung der Id's der genannten Tabellen

```
CREATE USER abteilungsleiter WITH PASSWORD 'prozess';
GRANT SELECT ON ALL TABLES IN SCHEMA {zustand1, zustand2, zustand3}
  TO abteilungsleiter;
GRANT INSERT, UPDATE, DELETE ON TABLE {zustand1, zustand2, zustand3}.lieferant
  TO abteilungsleiter;
GRANT INSERT, UPDATE, DELETE ON TABLE {zustand1, zustand2, zustand3}.ersatzteil
  TO abteilungsleiter;
GRANT INSERT, UPDATE, DELETE ON TABLE {zustand1, zustand2, zustand3}.lagerort
  TO abteilungsleiter;
```

Abteilungsleiter dürfen neue Ersatzteile und Lieferanten der Datenbank zufügen und diese den Lagern in der Tabelle Lagerort den Lagern zuordnen.

2.1.17 Funktionen

- *aktualisiert()* →aktualisiert den Wert in *letzte_aktualisierung*, sobald ein anderer Wert der Zeile verändert wurde (→Tabelle für Ersatzteile, Tabelle für Lieferanten)
- *teil_zugefuehrt()* →aktualisiert den Wert in *letzter_zugang* sobald der Wert 'anzahl' in der Zeile erhöht wurde
- *teil_entnommen()* →aktualisiert den Wert in *letzter_abgang* sobald der Wert 'anzahl' in der Zeile verringert wurde

```
CREATE FUNCTION aktualisiert() RETURNS TRIGGER
LANGUAGE plpgsql
AS $$
  BEGIN
  NEW.letzte_aktualisierung = CURRENT_TIMESTAMP;
  RETURN NEW
  END
   $$;

CREATE FUNCTION teil_zugefuehrt() RETURNS TRIGGER
```

```
LANGUAGE plpgsql
AS $$
  BEGIN
  IF (OLD.anzahl < NEW.ANZAHL) THEN NEW.letzter_zugang = CURRENT_TIMESTAMP;
  END IF;
  RETURN NEW;
  END
   $$;

CREATE FUNCTION teil_entnommen() RETURNS TRIGGER
LANGUAGE plpgsql
AS $$
  BEGIN
  IF (OLD.anzahl > NEW.anzahl) THEN NEW.letzter_abgang = CURRENT_TIMESTAMP;
  END IF;
  RETURN NEW;
  END
   $$;
```

2.2 Datenbankzustände

Zustand1 und Zustand2 werden beim ersten Starten der Datenbank sowie nach dem Löschen von */data* automatisch per SQL-Skript befüllt
Zustand1: *init/sample/zustand1.sql*
Zustand2: *init/sample/zustand2.sql*

In den Zuständen ist eine Beispielsituation von einem Unternehmen abgebildet (stark vereinfacht). Die Land-, Regierungsbezirk-, Stadt- und Standort-, Abteilung- und Lager-Tabellen dienen der späteren Auswertung und Reporterstellung, sodass sinnvolle Ergebnisse entstehen. Die in den Tabellen für Lieferanten, Ersatzteile, Lagerorte und Zuordnung(von Ersatzteilen zu Abteilungen) wird aktiv gearbeitet, um die Daten für die späteren Reports zu gewinnen sowie die Ersatzteilsituation zu managen.

Tabelle für Länder

```
land_id land_name
=====================================
DE        Deutschland
BE        Belgien
FR        Frankreich
AT        Österreich
CN        China
US        Vereinigte Staaten von Amerika
```

Beispielanfrage zum Einfügen:

```
INSERT INTO land(land_id, land_name) VALUES
  ('DE', 'Deutschland'), ... ;
```

Tabelle für Regierungsbezirke

regbez_id	land_id	regbez_name
DE:BW	DE	Baden-Württemberg
DE:BY	DE	Bayern
DE:BE	DE	Berlin
DE:BB	DE	Brandenburg
DE:HB	DE	Bremen
DE:HH	DE	Hamburg
DE:HE	DE	Hessen
DE:MV	DE	Mecklemburg-Vorpommern
DE:NI	DE	Niedersachsen
DE:NW	DE	Nordrhein-Westfahlen
DE:RP	DE	Reinland-Pfalz
DE:SL	DE	Saarland
DE:SN	DE	Sachsen
DE:ST	DE	Sachsen-Anhalt
DE:SH	DE	Schleswig-Holstein
DE:TH	DE	Thüringen
US:TX	US	Texas
US:VA	US	Virginia
CN-AH	CN	Anhui
CN-HA	CN	Henan
AT-5	AT	Salzburg
AT-2	AT	Kärnten
FR:BRE	FR	Bretagne
FR:IDF	FR	Ile-de-France
FR:COR	FR	Korsika
BE:WBR	BE	Provinz Wallonisch-Brabant
BE:VOV	BE	Provinz Ostflandern
BE:BRU	BE	Region Brüssel-Hauptstadt

Beispielanfrage zum Einfügen:

```
INSERT INTO regbez(regbez_id, land_id, regbez_name) VALUES
  ('DE:ST', 'DE', 'Sachsen-Anhalt'), ... ;
```

Tabelle für Städte

```
id      regbez_id    stadt_name                 plz
=================================================
1       DE:ST        Lutherstadt Wittenberg     06886
2       DE:ST        Halle(Saale)               06110
3       DE:ST        Halle(Saale)               06108
4       DE:BB        Potsdam                    14467
5       DE:BY        Nürnberg                   90403
6       DE:BY        Ingelfingen                74653
7       US:TX        Austin                     78652
8       US:TX        Houston                    77001
9       AT-2         Klagenfurt am Wörthersee   9010
10      AT-5         Salzburg                   5020
11      AT-5         Salzburg                   5082
12      DE:NW        Düsseldorf                 40468
13      DE:SH        Glinde                     21509
14      DE:BW        Weil am Rhein              79756
15      DE:NW        Essen                      45145
```

Beispielanfrage zum Einfügen:

```
INSERT INTO stadt(stadt_id, regbez_id, stadt_name, plz) VALUES
  (default, 'DE:ST', 'Lutherstadt Wittenberg)', '06886'), ... ;
```

2.2.1 Tabelle für Standorte

```
id  stadt   standort_name          anschrift
=============================================
10  1       Werk Wittenberg        Heuweg 5
11  2       Werk Halle I           Torstrae 1
12  3       Werk Halle II          Mansfelder Str. 11
20  6       Werk Austin 1          75th South 86
21  6       Werk Austin 2          76th South 155
31  8       Standort Klagenfurt    Salzburger Alee 142
```

Beispielanfrage zum Einfügen:

```
INSERT INTO standort(standort_id, stadt_id, standort_name, anschrift) VALUES
  ('10', 2, 'Werk Wittenberg', 'Torstrae 1'), ... ;
```

Tabelle für Abteilungen

```
abt_id    standort   abteilung_name
=================================
1001      10         Annahme
```

```
1051      10         CIP 1
1049      10         Versand
3109      31         Verpackung
1102      11         Abfuellung
1103      11         Trocknung
3151      31         CIP 1
3152      31         CIP 2
```

Beispielanfrage zum Einfügen:

```
INSERT INTO abteilung(abt_id, standort_id, abteilug_name) VALUES
  ('1001', '10', 'Annahme'), ... ;
```

Tabelle für Lager

```
id      standort  lager_name
=====================================
1081    10        Wittenberg L1
1082    10        Wittenberg L2
3181    31        Klagenfurt L1
1181    11        Halle 1L1
2181    21        Austin I Storage 1
```

Beispielanfrage zum Einfügen:

```
INSERT INTO lager(lager_id, standort_id, lager_name) VALUES
  ('1081', '10', 'Wittenberg L1'), ... ;
```

Tabelle für Lagerorte

a →anzahl mb→mindestbestand l_a →letzter_abgang l_z →letzter_zugang

```
id    e_id lager    a       mb      l_a                    l_z
=============================================================================================
1     1    1081     4       2        2020-08-01 14:34:22          \N
2     2    1081     11      5       \N                     2020-08-01 14:31:52
3     3    1081     1       1       \N                     \N
4     4    1081     7       5       \N                     \N
5     5    1081     8       7       \N                     \N
6     6    1081     4       4       \N                     \N
7     1    1082     7       5       \N                     \N
8     2    1082     3       2       \N                     \N
9     3    1082     5       4       \N                     \N
10    4    1082     7       6       \N                     \N
11    5    1087     7       5       \N                     \N
12    6    1082     4       3       \N                     \N
14    2    3181     3       2       \N                     \N
15    3    3181     5       4       \N                     \N
```

```
16    4    3181     7       6       \N                    \N
17    5    3181     7       5       \N                    \N
18    6    3181     4       3       \N                    \N
```

Beispielanfrage zum Einfügen:

```
INSERT INTO lagerort(lagerort_id, e_id, lager_id, anzahl, mindestbestand) VALUES
  (default, '1', '1081', 4, 2), ... ;
```

Tabelle für Zuordnung der Ersatzteile zu Abteilungen

```
e_id        abteilung_id
=========================
1            1001
1            1051
1            3101
2            3151
2            3152
3            1103
4            1001
5            1001
6            1001
```

Beispielanfrage zum Einfügen:

```
INSERT INTO zuordnung(e_id, abteilung_id) VALUES
  (1, '1001'), ... ;
```

Tabelle für Ersatzteile

l_id →lieferant_id

```
id  eclass        l_id   kennzecihnung  kosten   p_id   letzte_aktualisierung
=============================================================================
1   27-22-06-01   1      6213           149.95   a      2020-08-01 10:29:56
2   36-41-01-00   3      LKH            489.49   b      2020-08-01 10:29:56
2   36-10-01-04   2      CF-4000        629.99   c      2020-08-01 10:29:56
4   27-20-13-00   4      CerabarM       134.49   b      2020-08-01 10:29:56
5   27-20-13-00   4      CeraphanT      139.99   a      2020-08-01 10:29:56
6   27-27-31-00   5      SI2200         45.49    a      2020-08-01 10:29:56
7   37-01-02-03   6      KVS            104.95   b      2020-08-01 10:29:56
```

Beispielanfrage zum Einfügen:

```
INSERT INTO ersatzteil(e_id, eclass, l_id, kennzeichnung, kosten, p_id) VALUES
  (default, '27-22-06-01', 1, '6213', 149.95, 'a'), ... ;
```

Tabelle für Lieferanten

ap →ansprechpartner

id	lieferant_name	s_id	anschrift	email	ap	letzte_aktualisierung
1	Bürkert GmbH & Co.KG	6	Christian-Bürkert-Strae 13-17	vertrieb@bürkert.de	\N	2020-08-01 10:29:56
2	Gea Group Aktiengesellschaft	12	Peter-Müller-Strae 12	info@gea.com	Herr Müller	2020-08-01 10:29:56
3	Alfa Laval Mid Europe GmbH	13	Wilhelm-Bergner-Strae 7	info.mideurope@alfalaval.com	\N	2020-08-01 10:29:56
4	Endress+Hauser (Deutschland) GmbH+Co.KG	14	Colmarer Strae 6	info.de@endress.com	\N	2020-08-01 10:29:56
5	ifm electronic gmbh	15	Friedrichstrae 1	info@ifm.com	\N	2020-08-01 10:29:56
6	Flowserve Essen GmbH	15	Schederhofstr. 71	fcd@flowserve.com	\N	2020-08-01 10:29:56

Beispielanfrage zum Einfügen:

```
INSERT INTO lieferanten(lieferant_id, lieferant_name, stadt_id, anschrift,
   email, ansprechpartner) VALUES
  (default, 'Bürkert GmbH & CO.KG', 6, 'Christian-Bürkert-Strae 13-17',
  'vertrieb@bürkert.de', null), ... ;
```

Tabelle für eclass-Kategorisierung

eclass	beschreibung
36-41-03-05	Schraubenspindelpumpe
36-41-91-90	Dosierpumpe
36-41-01-00	Kreiselpumpe
27-22-06-01	Magnetventil
37-01-02-03	Regelventil
37-01-18-01	Membranventil
27-20-13-00	Druckaufnehmer
27-27-31-00	Stömungswächter
27-20-05-18	Füllstandsmessgerät
27-18-07-01	Klima-Schaltschrank
36-09-05-01	Druckluftfilter
22-41-11-00	Luftfilter für Lüftungssystem
27-24-22-00	Speicherprogrammierte Steuerung
36-43-04-03	Seitenkanalkompreossor
22-41-15-03	Kondensatpumpe
36-10-01-04	Dekantiergeräte (für Flüssigkeiten)

Beispielanfrage zum Einfügen:

```
INSERT INTO eclass(eclass, beschreibung) VALUES
  ('27-22-06-01', 'Magnetventil'), ... ;
```

Tabelle für Priorisierung der Lagersituation

p_id	beschreibung
a	Ersatzteil muss auf Lager sein

```
b       Lieferzeit zwischen einem Tag und einer Woche
c       Lieferzeit von mehr als einer Woche

INSERT INTO priorisierung(p_id, beschreibung) VALUES
  ('a', 'Ersatzteil muss auf Lager sein'), ... ;
```

2.2.2 Übergang Zustand 1 in Zustand 2

Die Datenbankzustände sind bis auf die Werte in der Spalte <Zustand>.lagerort.anzahl identisch. Der Unterschied könnte durch einen Systemausfall zustande gekommen sein, sodass in Zustand 2 weiter gebucht wurde während Zustand 1 nicht aktiv war. Anfrage, um alle 'anzahl'-Werte zu aktualisieren:

```
UPDATE zustand1.lagerort SET anzahl = z2.anzahl
FROM (SELECT * FROM zustand2.lagerort) as z2
WHERE lagerort.e_id = z2.e_id and lagerort.lager_id = z2.lager_id;
```

Diese Anfrage gewährleistet, dass unabhängig von der *Lagerort_id* die richtigen Werte den richtigen Ersatzteil-/Lager-Kombinationen zugewiesen wird.

2.2.3 */init/startup.sh*

Verwaltet den Startablauf

```
#/bin/bash

set -e

#Systemadministrator
nutzer="admin"
pw_nutzer="data"
recht_nutzer="SUPERUSER"

#Lagerist
nutzer1="lagerist"
pw_nutzer1="logistik"

#Abteilungsleiter
nutzer2="abteilungsleiter"
pw_nutzer2="prozess"

#Nutzerarray
declare -a nutzer=($nutzer $nutzer1 $nutzer2)
nutzer_length=${#nutzer[@]}
```

```
#Passwordarray
declare -a pw=($pw_nutzer $pw_nutzer1 $pw_nutzer2)
pw_length=${#pw[@]}
...
```

- Nutzer werden initialisiert
- in einem Array zum späteren Zugriff in Schleifen gespeichert

```
#Datenbank
db="etm"
db_full="Ersatzteilmanagement"

#Zustaende
zustand1="zustand1"
zustand2="zustand2"
zustand3="zustand3"
declare -a zustand=($zustand1 $zustand2 $zustand3)
zustand_length=${#zustand[@]}
```

- DB-Name und Schema Namen werden initialisiert
- Schemanamen werden in einem Array zum späteren Zugriff in Schleifen gespeichert

```
#Indexe
index1="anzahl_mindestbestand"
index2="kosten_eclass"
```

→Indexnamen werden festgelegt

```
echo ------------------------------------------------------------------------
echo Anzahl Zustaende: $zustand_length
echo Anzahl Nutzer: $nutzer_length
echo
echo Erstelle Datenbank $db "(Abkuerzung fuer $db_full)"
echo
psql -v ON_ERROR_STOP=1 --username "$POSTGRES_USER" --dbname "$POSTGRES_DB" << EOSQL
  CREATE DATABASE $db;
EOSQL
```

→Datenbank wird als User 'postgres' angelegt

```
#Schemaerstellung
echo Schemaerstellung
echo ---------------
for((i=0;i<$zustand_length;i++));
 do
   echo Erstelle ${zustand[$i]}
        psql -v ON_ERROR_STOP=1 --username "$POSTGRES_USER" --dbname "$POSTGRES_DB" << EOSQL
  \connect $db
  CREATE SCHEMA ${zustand[$i]};
  SET SEARCH_PATH TO ${zustand[$i]};
  \i docker-entrypoint-initdb.d/sample/create.sql
EOSQL
  echo ${zustand[$i]} erstellt.
  echo
 done
```

→Anlegen der Schemas, anschliessend Tabellenerstellung (über Schleife automatisch für alle 3 Zustände) →Befehle zur Tabellenerzeugung siehe *2.1.2-2.1.13*

```
#Indexerstellung
echo Erstelle Index
echo ----------------
for ((i=0;i<$nutzer_length;i++));
do
  psql -v ON_ERROR_STOP=1 --username "$POSTGRES_USER"  --dbname "$POSTGRES_DB" << EOSQL
  \connect $db
    CREATE INDEX $index1 ON ${zustand[$i]}.lagerort(anzahl, mindestbestand) ;
        CREATE INDEX $index2 ON ${zustand[$i]}.ersatzteil(kosten, eclass);
EOSQL
  echo
done
```

→Indexerstellung

```
#Nutzererstellung
echo Nutzererstellung
echo ----------------
for ((i=0;i<$nutzer_length;i++));
do
  echo Erstelle Nutzer: ${nutzer[$i]}
  psql -v ON_ERROR_STOP=1 --username "$POSTGRES_USER"  --dbname "$POSTGRES_DB" << EOSQL
    CREATE USER ${nutzer[$i]} WITH PASSWORD '${pw[$i]}';
EOSQL
  echo Nutzer: ${nutzer[$i]} erstellt.
  echo
done
#
#Rechtevergabe
echo Rechtevergabe
echo -------------
echo Allgemeine Rechte
echo
echo  Verteile allgemeine Rechte.
for ((i=0;i<$nutzer_length;i++));
do
  psql -v ON_ERROR_STOP=1 --username "$POSTGRES_USER" --dbname "$POSTGRES_DB" << EOSQL
    \connect $db
  GRANT CONNECT ON DATABASE $db to ${nutzer[$i]};
  GRANT pg_read_server_files TO ${nutzer[$i]};
  GRANT USAGE ON SCHEMA ${zustand[0]} TO ${nutzer[$i]};
  GRANT USAGE ON SCHEMA ${zustand[1]}  TO ${nutzer[$i]};
  GRANT USAGE ON SCHEMA ${zustand[2]}  TO ${nutzer[$i]};
EOSQL
done
echo Allgemeine Rechte verteilt
echo
echo Spezifische Rechte
echo
echo Verteile spezifische Rechte für ${nutzer[0]}
for ((i=0;i<$zustand_length;i++));
do
  psql -v ON_ERROR_STOP=1 --username "$POSTGRES_USER" --dbname "$POSTGRES_DB" << EOSQL
               \connect $db
  ALTER USER ${nutzer[0]} WITH $recht_nutzer;
  GRANT ALL PRIVILEGES ON SCHEMA ${zustand[$i]} TO ${nutzer[1]};
EOSQL
done
echo Spezifische Rechte für ${nutzer[0]} verteilt
echo
echo Verteile spezifische Rechte für ${nutzer[1]}
for ((i=0;i<$zustand_length;i++));
do
  psql -v ON_ERROR_STOP=1 --username "$POSTGRES_USER" --dbname "$POSTGRES_DB" << EOSQL
               \connect $db
  GRANT SELECT ON ALL TABLES IN SCHEMA ${zustand[$i]} TO ${nutzer[1]};
  REVOKE SELECT (kosten) on TABLE ${zustand[$i]}.ersatzteil FROM ${nutzer[1]};
  GRANT UPDATE (anzahl) ON TABLE ${zustand[$i]}.lagerort TO ${nutzer[1]};
EOSQL
done
echo Spezifische Rechte für ${nutzer[1]} verteilt
echo
echo Verteile spezifische Rechte für ${nutzer[1]}
for ((i=0;i<$zustand_length;i++));
do
  psql -v ON_ERROR_STOP=1 --username "$POSTGRES_USER" --dbname "$POSTGRES_DB" << EOSQL
               \connect $db
  GRANT SELECT ON ALL TABLES IN SCHEMA ${zustand[$i]} TO ${nutzer[2]};
  GRANT INSERT, UPDATE,  DELETE ON TABLE ${zustand[$i]}.lieferant TO ${nutzer[2]};
```

```
  GRANT INSERT, UPDATE,  DELETE ON TABLE ${zustand[$i]}.ersatzteil TO ${nutzer[2]};
  GRANT INSERT, UPDATE,  DELETE ON TABLE ${zustand[$i]}.lagerort TO ${nutzer[2]};
  GRANT ALL PRIVILEGES ON SEQUENCE ${zustand[$i]}.lieferant_id_seq TO ${nutzer[2]};
  GRANT ALL PRIVILEGES ON SEQUENCE ${zustand[$i]}.e_id_seq TO ${nutzer[2]};
  GRANT ALL PRIVILEGES ON SEQUENCE ${zustand[$i]}.lagerort_id_seq TO ${nutzer[2]};
EOSQL
done
echo Spezifische Rechte für ${nutzer[2]} verteilt
```

- Nutzererstellung als Schleife

→ iteriert über das nutzerArray und passwordArray und liest daraus die Namen und Passwörter aus

- Rechtevergabe erfolgt zunächst allgemein, d.h. über eine Schleife werden die allgemeinen Rechte an alle Nutzer vergeben
- Die Vergabe spezifischer Rechte erfolgt über eine Schleife, die über das ßustand-Array iteriert, sodass die Anfrage zur Rechtevergabe auf einem Schema nur einmal statt 3x im Script steht

```
echo Zustandsbefuellung
echo ------------------
echo
echo Befuele ${zustand[0]}
psql -v ON_ERRROR_STOP=1 --username "$POSTGRES_USER" --dbname "$POSTGRES_DB" << EOSQL
\connect $db
SET SEARCH_PATH TO ${zustand[0]};
\i docker-entrypoint-initdb.d/sample/zustand_1.sql
EOSQL
echo ${zustand[0]} befuellt.
echo
echo Befuele ${zustand[1]}
psql -v ON_ERROR_STOP=1 --username "$POSTGRES_USER" --dbname "$POSTGRES_DB" << EOSQL
\connect $db
SET SEARCH_PATH TO ${zustand[1]};
\i docker-entrypoint-initdb.d/sample/zustand_2.sql;
EOSQL
echo ${zustand[2]} befuellt.
echo Befuele ${zustand[2]}
psql -v ON_ERROR_STOP=1 --username "$POSTGRES_USER" --dbname "$POSTGRES_DB" << EOSQL
\connect $db
SET SEARCH_PATH TO ${zustand[2]};
--\i docker-entrypoint-initdb.d/sample/zustand_3.sql;
EOSQL
echo ${zustand[2]} befuellt.
echo ------------------------------------------------------------------------
```

- Zustandsbefüllung erfolgt über den Aufruf von SQL-Sktipten (Anweisungen in *2.1.2-2.1.13*

2.2.4 Logausgabe

(Schleifen auf eine Iteration gekürtzt)

```
------------------------------------------------------------------------
Anzahl Zustaende: 3
Anzahl Nutzer: 3

Erstelle Datenbank etm (Abkuerzung fuer Ersatzteilmanagement)

CREATE DATABASE
```

```
Schemaerstellung
----------------
Erstelle zustand1
You are now connected to database "etm" as user "postgres".
CREATE SCHEMA
SET
CREATE FUNCTION
CREATE TABLE
CREATE TABLE
CREATE SEQUENCE
CREATE TABLE
CREATE TABLE
CREATE TABLE
CREATE TABLE
CREATE SEQUENCE
CREATE TABLE
CREATE TRIGGER
CREATE TABLE
CREATE TABLE
INSERT 0 3
CREATE SEQUENCE
CREATE TABLE
CREATE TRIGGER
CREATE TABLE
CREATE SEQUENCE
CREATE TABLE
CREATE FUNCTION
CREATE TRIGGER
CREATE FUNCTION
CREATE TRIGGER
zustand1 erstellt.

Erstelle zustand2
You are now connected to database "etm" as user "postgres".
CREATE SCHEMA
...
...
...
CREATE TRIGGER
zustand2 erstellt.

Erstelle zustand2
You are now connected to database "etm" as user "postgres".
CREATE SCHEMA
...
...
...
CREATE TRIGGER
```

```
zustand3 erstellt.

Erstelle Index
--------------
You are now connected to database "etm" as user "postgres".
CREATE INDEX
CREATE INDEX
You are now connected to database "etm" as user "postgres".
CREATE INDEX
CREATE INDEX
You are now connected to database "etm" as user "postgres".
CREATE INDEX
CREATE INDEX

Nutzererstellung
----------------
Erstelle Nutzer: admin
CREATE ROLE
Nutzer: admin erstellt.

Erstelle Nutzer: lagerist
CREATE ROLE
Nutzer: abteilungsleiter erstellt.

Erstelle Nutzer: abteilungsleiter
CREATE ROLE
Nutzer: abteilungsleiter erstellt.

Rechtevergabe
--------------
Allgemeine Rechte

Verteile allgemeine Rechte
You are now connected to database "etm" as user "postgres".
GRANT
GRANT ROLE
GRANT
GRANT
GRANT
You are now connected to database "etm" as user "postgres".
...
You are now connected to database "etm" as user "postgres".
GRANT
GRANT ROLE
GRANT
GRANT
GRANT
Allgemeine Rechte verteilt
```

```
Spezifische Rechte

Verteile spezifische Rechte für admin
You are now connected to database "etm" as user "postgres".
ALTER ROLE
GRANT
...
Spezifische Rechte für admin verteilt

Verteile spezifische Rechte für lagerist
You are now connected to database "etm" as user "postgres".
GRANT
REVOKE
GRANT
Spezifische Rechte für lagerist verteilt

Verteile speziische für abteilungsleiter
You are now connected to database "etm" as user "postgres".
GRANT
GRANT
GRANT
GRANT
GRANT
GRANT
GRANT
You are now connected to datatbase "etm" as user "postgres".
...
Spezifische Rechte für abteilungsleiter verteilt

Zustandsbefuellung
------------------
befuelle zustand1
You are now connected to datatbase "etm" as user "postgres".
SET
INSERT 0 6
INSERT 0 28
INSERT 0 15
INSERT 0 6
INSERT 0 8
INSERT 0 5
INSERT 0 16
INSERT 0 6
INSERT 0 7
INSERT 0 9
INSERT 0 18
zustand 1 befuellt.

befuelle zustand2
...
```

```
zustand2 befuellt
befuelle zustand3
You are now connected to datatbase "etm" as user "postgres".
SET
zustand3 befuellt
------------------------------------------------------------------------
```

3 Anwendungsprogramme

3.1 Maven-Projekt mit Abhängigkeiten

Anwendung1.jar → *SELECT / UPDATE*

Anwendung2.jar → *INSERT / DELETE*

3.1.1 Abhängkeiten / Plugins

1. JDBC-Treiber für Java

   GroupId: *org.postgresql*

   ArtifactId: *postgresql*

   Version: *42.2.14*

   → ermöglicht die Verbindung mit einer postgreSQL-Datenbank

2. Maven-Jar-Plugin

   GroupId: *org.apache.maven.plugins*

   ArtifactId: *maven-jar-plugin*

   Version: *3.20*

   → ermöglicht das Bauen von ausführbaren *.jar-Anwendungen

3. Maven-Assembly-Plugin

   GroupId: *org.apache.maven.plugins*

   ArtifactId: *maven-assembly-plugin*

   → fügt den ausführbaren *.jar-Anwendungen die benötigten Abhängigkeiten hinzu

→ Befehl zum Bauen der jar-Anwendungen:
mvn clean compile pacakge assembly:single

3.1.2 Maven-Build-Log

```
[INFO] Scanning for projects...
[INFO]
[INFO] ---------------< mlu.ajtrs.dbp20-projekt:dbp20-projekt >----------------
[INFO] Building dbp20-projekt 0.0.1-SNAPSHOT
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ dbp20-projekt ---
[INFO] Deleting /home/jonathan/eclipse-workspace/dbp20/target
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ dbp20-projekt ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 0 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ dbp20-projekt ---
[INFO] Changes detected - recompiling the module!
[WARNING] File encoding has not been set, using platform encoding UTF-8, i.e. build is platform dependent!
[INFO] Compiling 1 source file to /home/jonathan/eclipse-workspace/dbp20/target/classes
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ dbp20-projekt ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 0 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ dbp20-projekt ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ dbp20-projekt ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 0 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ dbp20-projekt ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ dbp20-projekt ---
[INFO]
[INFO] --- maven-jar-plugin:3.2.0:jar (default-jar) @ dbp20-projekt ---
[INFO] Building jar: /home/jonathan/eclipse-workspace/dbp20/target/dbp20-projekt-0.0.1-SNAPSHOT.jar
[INFO]
[INFO] --- maven-assembly-plugin:2.2-beta-5:single (default-cli) @ dbp20-projekt ---
[INFO] Building jar: /home/jonathan/eclipse-workspace/dbp20/target/dbp20-projekt-0.0.1-SNAPSHOT-jar-with-dependencies.jar
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  16.103 s
[INFO] Finished at: 2020-08-01T15:23:57+02:00
[INFO] ------------------------------------------------------------------------
```

3.2 CRUD-Anwendungsprogramm

3.2.1 Anwendung1.jar →SELECT, UPDATE

Aufgabe Aktualisierung der Ersatteilanzahl in den Lagern

Aufruf *java -jar Anwendung1.jar* (in der Konsole)

Ablauf

1. Verbinden mit Datenbank per JDBC
1.1 AutoComit auf *false* setzen

```
select lager_id as Lager_Id, lager_name as Name from zustand1.lager;
```

2. Abfrage, in welchem Lager Änderungen vorgenommen werden sollen (lager_id →?)

```
select lo.anzahl as ab, lo.mindestbestand as mb, e.kennzeichnung, l.lieferant_name,
      e.e_id as id, lo.letzter_abgang, lo.letzter_zugang
from zustand1.lagerort lo join zustand1.ersatzteil e on (e.e_id = lo.e_id)
                          join zustand1.lieferant l on (e.lieferant_id = l.lieferant_id)
where lager_id = ?;
```

3. Abfrage, welches Ersatzteil bearbeitet werden soll (e_id)
4. Abfrage, auf welchen Wert 'anzahl' gesetzt werden soll (anzahl)

```
update zustand1.lagerort set anzahl = ? where e_id = ?;
```

5. Abfrage, ob Änderungen bestätigt (commited) werden sollen.

```
select lo.anzahl as ab, lo.mindestbestand as mb, e.kennzeichnung, l.lieferant_name,
      e.e_id as id, lo.letzter_abgang, lo.letzter_zugang
from zustand1.lagerort lo join zustand1.ersatzteil e on (e.e_id = lo.e_id)
                          join zustand1.lieferant l on (e.lieferant_id = l.lieferant_id)
where lager_id = ?;
```

5.1 **ja** : Commit wird ausgeführt, anschliessend beendet das Programm →Werte werden in Datenbank gespeichert
5.2 **nein** : Programm beendet ohne Commit →Werte werden in Datenbank →Werte werden in Datenbank nicht gespeichert

3.2.2 Beispielablauf

```
Ersatzteillager-Aktualisierung
------------------------------
Erfolgreich mit Datenbank verbunden.
Auto-Commit Modus: false

lager_id|name
1081        |Wittenberg L1
1082        |Wittenberg L2
3181        |Klagenfurt L1
1181        |Halle 1L1
2181        |Austin I Storage 1
(UPDATE-Beispiel)
In welchem Lager möchten Sie Änderungen vornehmen?
(lager_id) 1081

Aktuelle Situation im Lager 1081
ab = aktueller Bestand, mb = Mindestbestand
    id |  ab |  mb |                      kennzeichnung |                                lieferant_name |         letzter_abgang |         letzter_zugang
     1 |   4 |   2 |                              6213 |                           Bürkert GmbH & Co.KG |                   null |                   null
     2 |  11 |   5 |                               LKH |                     Alfa Laval Mid Europe GmbH |                   null |                   null
     3 |   1 |   1 |                           CF-4000 |                Gea Group Aktiengesellschaft |                   null |                   null
     4 |   7 |   5 |                          CerabarM |          Endress+Hauser (Deutschland) GmbH+Co.KG |                   null |                   null
     5 |   8 |   7 |                         CeraphanT |          Endress+Hauser (Deutschland) GmbH+Co.KG |                   null |                   null
     6 |   4 |   4 |                            SI2200 |                      ifm electronic gmbh |                   null |                   null


Möchten sie die Anzahl eines bestehenden Ersatzteils ändern?
(j|n)j

Um welches Ersatzteil handelt es sich?
(e_id)1
Änderung des Bestands auf: 5

Aktuelle Situation im Lager 1081
ab = aktueller Bestand, mb = Mindestbestand
    id |  ab |  mb |                      kennzeichnung |                                lieferant_name |         letzter_abgang |         letzter_zugang
     1 |   5 |   2 |                              6213 |                           Bürkert GmbH & Co.KG |                   null |    2020-08-01 20:02:46
     2 |  11 |   5 |                               LKH |                     Alfa Laval Mid Europe GmbH |                   null |                   null
     3 |   1 |   1 |                           CF-4000 |                Gea Group Aktiengesellschaft |                   null |                   null
     4 |   7 |   5 |                          CerabarM |          Endress+Hauser (Deutschland) GmbH+Co.KG |                   null |                   null
     5 |   8 |   7 |                         CeraphanT |          Endress+Hauser (Deutschland) GmbH+Co.KG |                   null |                   null
     6 |   4 |   4 |                            SI2200 |                      ifm electronic gmbh |                   null |                   null


Änderungen bestätigen?
(j|n)
j
Änderungen bestätigt.
Erfolgreich von der Datenbank getrennt.
```

3.2.3 Anwendung2.jar →INSERT, DELETE

Aufgabe Aktualisierung der Lieferanten und Ersatzteiltabellen

Aufruf *java -jar Anwendung2.jar* (in der Konsole)

Ablauf INSERT

1. Verbinden mit Datenbank per JDBC
1.1 AutoComit auf *false* setzen

```
select l.lieferant_id as id, l.lieferant_name as name, l.email, l.ansprechpartner
from zustand1.lieferant l order by l.lieferant_id asc;
```

2. Frage, ob Lieferant hinzugefügt werden soll 3. Eingabe der Daten des Lieferanten →Vorbereitung für den Insert

```
select * from zustand1.stadt;
```

3.1 Frage nach der Stadt des Lieferanten 3.2 aktuellen Sequenzwert speichern

```
INSERT INTO zustand1.lieferant(lieferant_id, lieferant_name,stadt_id,
   anschrift, email, ansprechpartner) VALUES (default, ?, ?, ?, ?, ?);
```

4. abgefragte Werte einsetzten und Update durchführen

```
select l.lieferant_name, l.anschrift, s.stadt_name, s.plz, l.email, l.ansprechpartner
from zustand1.lieferant l join zustand1.stadt s on (s.stadt_id = l.stadt_id)
where lieferant_id = currval('zustand1.lieferant_id_seq');
```

5. Ausgabe der eingefügten Werte und Frage nach Bestätigung des Inserts

```
select l.lieferant_id as id, l.lieferant_name as name, l.email, l.ansprechpartner
from zustand1.lieferant l order by l.lieferant_id asc;
```

6.1 **ja** : Commit wird ausgeführt, anschliessend beendet das Programm →Werte werden in Datenbank gespeichert
6.2 **nein** : Programm beendet ohne Commit →Werte werden in Datenbank nicht gespeichert

3.2.4 Beispielablauf INSERT

```
Ersatzteil- und Lieferantenverwaltung
--------------------------------------
Erfolgreich mit Datenbank verbunden.
Auto-Commit Modus: false

Lieferanten

  id |                                      name |                                    email |                                ansprechpart
   1 |                          Bürkert GmbH & Co.KG |                          vertrieb@bürkert.de |                                           n
   2 |                       Gea Group Aktiengesellschaft |                             info@gea.com |                                 Herr Mül
   3 |                         Alfa Laval Mid Europe GmbH |                 info.mideurope@alfalaval.com |                                           n
   4 |           Endress+Hauser (Deutschland) GmbH+Co.KG |                          info.de@endress.com |                                           n
   5 |                               ifm electronic gmbh |                             info@ifm.com |                                           n
   6 |                            Flowserve Essen GmbH |                           fcd@flowserve.com |                                           n


(INSERT-Beispiel)
Wollen sie einen Lieferanten hinzufügen?
(j|n)j
Name:
Test
```

```
Städte
(Stadt_Id, Regierungsbezirk_Id, Name, Plz


1        |DE:ST        |06886        |Lutherstadt Wittenberg
2        |DE:ST        |06110        |Halle(Saale)
3        |DE:ST        |06108        |Halle(Saale)
4        |DE:BB        |14467        |Potsdam
5        |DE:BY        |90403        |Nürnberg
6        |DE:BY        |74653        |Ingelfingen
7        |US:TX        |78652        |Austin
8        |US:TX        |77001        |Houston
9        |AT-2        |9010        |Klagenfurt am Wörthersee
10        |AT-5        |5020        |Salzburg
11        |AT-5        |5082        |Salzburg
12        |DE:NW        |40468        |Düsseldorf
13        |DE:SH        |21509        |Glinde
14        |DE:BW        |79756        |Weil am Rhein
15        |DE:NW        |45145        |Essen

Ist die Stadt in der Datenbank vorhanden?
(j|n)j
Stadt_Id: 3
Anschrift: Teststrae 999
E-Mail: test@test.de
Ansprechpartner: Test

Ihre Eingaben:
Name:                  Test
Anschrift:         Teststrae 999
Stadt:                 Halle(Saale)
PLZ:                06108
Email:                 test@test.de
Ansprechpartner: Test

Eingaben bestätigen?
(j|n)
j
Lieferanten

  id |                                          name |                                  email |                                ansprechpart
   1 |                              Bürkert GmbH & Co.KG |                      vertrieb@bürkert.de |                                          n
   2 |                     Gea Group Aktiengesellschaft |                             info@gea.com |                                  Herr Mü
   3 |                       Alfa Laval Mid Europe GmbH |          info.mideurope@alfalaval.com |                                          n
   4 |           Endress+Hauser (Deutschland) GmbH+Co.KG |                  info.de@endress.com |                                          n
   5 |                               ifm electronic gmbh |                         info@ifm.com |                                          n
   6 |                              Flowserve Essen GmbH |                    fcd@flowserve.com |                                          n
   7 |                                              Test |                         test@test.de |                                          1

Einfügen bestätigt.
Anwendung wird beendet. Von Datenbank trennen...
Erfolgreich von Datenbank getrennt.
```

→adäquater Abblauf beim Hinzufügen eines Ersatzteils

Ablauf DELETE

1. Verbinden mit Datenbank per JDBC
1.1 AutoComit auf *false* setzen

```
select l.lieferant_id as id, l.lieferant_name as name, l.email, l.ansprechpartner
from zustand1.lieferant l order by l.lieferant_id asc;
```

2. Frage welcher Lieferant gelöscht werden soll

```
select e.kennzeichnung, e.kosten, ec.eclass_beschreibung
from zustand1.lieferant l join zustand1.ersatzteil e on (e.lieferant_id = l.lieferant_id)
                          join zustand1.eclass ec on (ec.eclass = e.eclass)
where l.lieferant_id = ?;
```

3. Ausgabe der beteiligten Ersatzteile

```
delete from zustand1.ersatzteil where kennzeichnung = ?;
```

4. Frage nach bestätigen des Löschen

```
select l.lieferant_id as id, l.lieferant_name as name, l.email, l.ansprechpartner
from zustand1.lieferant l order by l.lieferant_id asc;
```

4.1 **ja** : Commit wird ausgeführt, anschliessend beendet das Programm →Werte werden in Datenbank gespeichert
4.2 **nein** : Programm beendet ohne Commit →Werte werden in Datenbank nicht gespeichert

```
Ersatzteil- und Lieferantenverwaltung
--------------------------------------
Erfolgreich mit Datenbank verbunden.
Auto-Commit Modus: false

Lieferanten

  id |                                        name |                                        email |                                ansprechpart
   1 |                              Bürkert GmbH & Co.KG |                              vertrieb@bürkert.de |                                        n
   2 |                     Gea Group Aktiengesellschaft |                                     info@gea.com |                                Herr Mü
   3 |                       Alfa Laval Mid Europe GmbH |                   info.mideurope@alfalaval.com |                                        n
   4 |           Endress+Hauser (Deutschland) GmbH+Co.KG |                              info.de@endress.com |                                        n
   5 |                               ifm electronic gmbh |                                     info@ifm.com |                                        n
   6 |                              Flowserve Essen GmbH |                                fcd@flowserve.com |                                        n
   7 |                                              Test |                                     test@test.de |                                        1


(INSERT-Beispiel)
Wollen sie einen Lieferanten hinzufügen?
(j|n)n

(DELETE-Beispiel)
Wollen Sie einen Lieferanten löschen?
(j|n)j
Lieferanten

  id |                                        name |                                        email |                                ansprechpart
   1 |                              Bürkert GmbH & Co.KG |                              vertrieb@bürkert.de |                                        n
   2 |                     Gea Group Aktiengesellschaft |                                     info@gea.com |                                Herr Mü
   3 |                       Alfa Laval Mid Europe GmbH |                   info.mideurope@alfalaval.com |                                        n
   4 |           Endress+Hauser (Deutschland) GmbH+Co.KG |                              info.de@endress.com |                                        n
   5 |                               ifm electronic gmbh |                                     info@ifm.com |                                        n
   6 |                              Flowserve Essen GmbH |                                fcd@flowserve.com |                                        n
   7 |                                              Test |                                     test@test.de |                                        1


Welcher Lieferant (inkl. Ersatzteile) soll gelöscht werden?
(lieferant_id) 4
Zugehörige Ersatzteile:

                                  kennzeichnung |      kosten |                          eclass_beschreibung
                                      CeraphanT |      139,99 |                               Druckaufnehmer
                                       CerabarM |      134,49 |                               Druckaufnehmer

Löschen bestätigen:
(j|n)
j
Lieferanten

  id |                                        name |                                        email |                                ansprechpart
   1 |                              Bürkert GmbH & Co.KG |                              vertrieb@bürkert.de |                                        n
   2 |                     Gea Group Aktiengesellschaft |                                     info@gea.com |                                Herr Mü
   3 |                       Alfa Laval Mid Europe GmbH |                   info.mideurope@alfalaval.com |                                        n
   5 |                               ifm electronic gmbh |                                     info@ifm.com |                                        n
   6 |                              Flowserve Essen GmbH |                                fcd@flowserve.com |                                        n
   7 |                                              Test |                                     test@test.de |                                        1

Löschen bestätigt.

Anwendung wird beendet. Von Datenbank trennen...
Erfolgreich von Datenbank getrennt.
```

→adäquater Abblauf beim Löschen eines Ersatzteils

3.3 Rekursive Anfrage (Bash-Skript)

Aufgabe Ausgabe der den Abteilungen an Standort '10' zugeordneten Lieferanten

Aufruf *bash Anwendung1.sh* (in der Konsole)

3.3.1 Relevanter Teil des DB-Zustands

Standort

```
id  stadt   standort_name          anschrift
=========================================================
10  1       Werk Wittenberg        Heuweg 5
```

Abteilung

```
abt_id    standort   abteilung_name
=========================================
1001      10         Annahme
1051      10         CIP 1
1049      10         Versand
```

Zuordnung

```
e_id        abteilung_id
=========================
1             1001
1             1051
4             1001
5             1001
6             1001
```

Ersatzteil

```
id  eclass        l_id   kennzecihnung  kosten   p_id   letzte_aktualisierung
=============================================================================
1   27-22-06-01   1      6213           149.95   a      2020-08-01 10:29:56
2   36-41-01-00   3      LKH            489.49   b      2020-08-01 10:29:56
2   36-10-01-04   2      CF-4000        629.99   c      2020-08-01 10:29:56
4   27-20-13-00   4      CerabarM       134.49   b      2020-08-01 10:29:56
5   27-20-13-00   4      CeraphanT      139.99   a      2020-08-01 10:29:56
6   27-27-31-00   5      SI2200         45.49    a      2020-08-01 10:29:56
7   37-01-02-03   6      KVS            104.95   b      2020-08-01 10:29:56
```

Lieferant

```
id lieferant_name                          s_id anschrift                    email                          ap          letzte_aktualisierung
=================================================================================================================================================
1  Bürkert GmbH & Co.KG                    6    Christian-Bürkert-Strae 13-17  vertrieb@bürkert.de            \N          2020-08-01 10:29:56
2  Gea Group Aktiengesellschaft             12   Peter-Müller-Strae 12          info@gea.com                   Herr Müller 2020-08-01 10:29:56
3  Alfa Laval Mid Europe GmbH               13   Wilhelm-Bergner-Strae 7        info.mideurope@alfalaval.com   \N          2020-08-01 10:29:56
4  Endress+Hauser (Deutschland) GmbH+Co.KG  14   Colmarer Strae 6               info.de@endress.com            \N          2020-08-01 10:29:56
5  ifm electronic gmbh                      15   Friedrichstrae 1               info@ifm.com                   \N          2020-08-01 10:29:56
6  Flowserve Essen GmbH                     15   Schederhofstr. 71              fcd@flowserve.com              \N          2020-08-01 10:29:56
```

3.3.2 SQL-Anfrage

```
with recursive max as (
        select max(lieferant_id) as max
        from lieferant
),
rec as (
        select distinct a.abteilung_id, cast(l.lieferant_id as integer), l.lieferant_name
         from lieferant l join ersatzteil e on (e.lieferant_id = l.lieferant_id)
            join zuordnung z on (z.e_id = e.e_id)
            join abteilung a on (a.abteilung_id = z.abteilung_id)
            join standort s on (s.standort_id = a.standort_id)
        where s.standort_id = '10' and l.lieferant_id = 1
        union
        select distinct a.abteilung_id, cast(l.lieferant_id+1 as integer), l.lieferant_name
         from lieferant l join ersatzteil e on (e.lieferant_id = l.lieferant_id)
            join zuordnung z on (z.e_id = e.e_id)
            join abteilung a on (a.abteilung_id = z.abteilung_id)
            join standort s on (s.standort_id = a.standort_id), max m, rec r
        where s.standort_id = '10' and l.lieferant_id<m.max)
select a.abteilung_name as "Abteilung", r.abteilung_id as "ID", string_agg(distinct r.lieferant_name, ', ') as "Lieferanten"
from rec r join abteilung a on (a.abteilung_id = r.abteilung_id)
group by r.abteilung_id, a.abteilung_name order by r.abteilung_id asc;
```

3.3.3 Ergebnis

```
 Abteilung |  ID  |                                  Lieferanten
-----------+------+-------------------------------------------------------------------------------
 Annahme   | 1001 | Bürkert GmbH & Co.KG, Endress+Hauser (Deutschland) GmbH+Co.KG, ifm electronic gmbh
 CIP 1     | 1051 | Bürkert GmbH & Co.KG
```

3.4 Report →ROLLUP (Bash-Skript)

Aufgabe Darstellung des Werts aller gelagerten Ersatzteile

Aufruf *bash rollup.sh* (in der Konsole)

3.4.1 Relevanter Teil des DB-Zustands

Standorte

id	stadt	standort_name	anschrift
10	1	Werk Wittenberg	Heuweg 5
11	2	Werk Halle I	Torstrae 1
12	3	Werk Halle II	Mansfelder Str. 11
20	6	Werk Austin 1	75th South 86
21	6	Werk Austin 2	76th South 155
31	8	Standort Klagenfurt	Salzburger Alee 142

Lager

id	standort	lager_name
1081	10	Wittenberg L1
1082	10	Wittenberg L2
3181	31	Klagenfurt L1
1181	11	Halle 1L1
2181	21	Austin I Storage 1

Lagerorte a →Anzahl, mb →Mindestbestand, l_a →letzter Abgang, l_b →letzter Zugang

id	e_id	lager	a	mb	l_a	l_z
1	1	1081	4	2	2020-08-01 14:34:22	\N
2	2	1081	11	5	\N	2020-08-01 14:31:52
3	3	1081	1	1	\N	\N
4	4	1081	7	5	\N	\N
5	5	1081	8	7	\N	\N
6	6	1081	4	4	\N	\N
7	1	1082	7	5	\N	\N
8	2	1082	3	2	\N	\N
9	3	1082	5	4	\N	\N
10	4	1082	7	6	\N	\N
11	5	1087	7	5	\N	\N
12	6	1082	4	3	\N	\N
14	2	3181	3	2	\N	\N
15	3	3181	5	4	\N	\N
16	4	3181	7	6	\N	\N
17	5	3181	7	5	\N	\N
18	6	3181	4	3	\N	\N

Ersatzteile

id	eclass	l_id	kennzecihnung	kosten	p_id	letzte_aktualisierung
1	27-22-06-01	1	6213	149.95	a	2020-08-01 10:29:56
2	36-41-01-00	3	LKH	489.49	b	2020-08-01 10:29:56
2	36-10-01-04	2	CF-4000	629.99	c	2020-08-01 10:29:56
4	27-20-13-00	4	CerabarM	134.49	b	2020-08-01 10:29:56
5	27-20-13-00	4	CeraphanT	139.99	a	2020-08-01 10:29:56
6	27-27-31-00	5	SI2200	45.49	a	2020-08-01 10:29:56
7	37-01-02-03	6	KVS	104.95	b	2020-08-01 10:29:56

3.4.2 SQL-Anfrage

```
select s.standort_name as "Standort", l.lager_name as "Lagername",
       sum(anzahl*kosten) || ' 'as "Wert aller gelagerten Ersatzteile"
from standort s join lager l on (s.standort_id = l.standort_id)
                join lagerort lo on (l.lager_id = lo.lager_id)
                join ersatzteil e on (e.e_id = lo.e_id)
group by rollup(s.standort_name, l.lager_name);
```

3.4.3 Ergebnis

```
      Standort       |   Lagername   | Wert aller gelagerten Ersatzteile
---------------------+---------------+-----------------------------------
                     |               | 18046.35
```

```
 Standort Klagenfurt | Klagenfurt L1 | 5550.13
 Werk Wittenberg     | Wittenberg L2 | 5550.13
 Werk Wittenberg     | Wittenberg L1 | 6946.09
 Werk Wittenberg     |               | 12496.22
 Standort Klagenfurt |               | 5550.13
```

4 Indexe

4.1 Index 1

```
CREATE INDEX kosten_eclass ON ersatzteil(kosten, eclass)

SELECT  e.e_id, e.kennzeichnung, l.lieferant_name,
        e.kosten, ec.eclass_beschreibung
FROM ersatzteil e join lieferant l on (l.lieferant_id = e.lieferant_id)
                  join eclass ec on (ec.eclass = e.eclass)
ORDER BY eclass_beschreibung;
```

→Eventuelle Kostenoptimierung durch Substition von Ersatzteilen von günstigeren Anbietern

4.2 Index 2

```
CREATE INDEX anzahl_mindestbestand ON lagerort(anzahl, mindestbestand);
--1.Anfrage
SELECT lagerort_id, lager_id, e_id, (lo.anzahl-lo.mindesbestand) as diff
FROM   lagerort lo
ORDER BY diff desc;
--2.Anfrage
SELECT l.lieferant_name, e.kennzeichnung, l.lager_id, sum(anzahl*kosten)
FROM  ersatzteil e join lagerort lo on (e.e_id = lo.e_id)
                   join lager l on (l.lager_id = lo.lager_id)
                   join lieferant li on (e.lieferant_id = li.lieferant_id)
GROUP BY e.e_id, l.lager_name;
1. Anfrage Optimierung der Lagerhaltung durch Abbau eventuell zu groer Bestände
2. Anfrage Berechnung der Werte und Zuordnung zu gelagerten Ersatzteilen
```

5 Entwicklung

5.1 Git-Historie

```
* 09d075a (HEAD -> master, origin/master, origin/HEAD) (Quellcode und pom.xml für die Java Anwendungen, 2020-07-31)
* 0e011b8 (Rechte aktualisiert, 2020-07-31)
* bef6c04 (zustand3, 2020-07-31)
* c223070 (Rechte für Lagerist und Abteilungsleiter überarbeitet, 2020-07-31)
*   012a28b (Merge branch 'master' of https://gitlab-bs.kube.informatik.uni-halle.de/ajtrs/dbp20-projekt, 2020-07-31)
|\
| * 09590f7 (Formatierung, 2020-07-31)
```

```
* | 775f905 (lageist-Recht  bearbeitet | Indexe hinzugefügt, 2020-07-31)
|/
* 74c935a (readme, 2020-07-31)
* 8fef9c8 (readme erweitert, 2020-07-31)
* 0d2bdcb (zustand 2 unterscheidet sich nur in anzahl int lagerort tabelle mit zustand 1, 2020-07-31)
* 6c1efa0 (bash Skripte, 2020-07-31)
* a2165d7 (DW-Rollup hinzugefügt, 2020-07-31)
* 21a3a16 (start-skript aktualisiert, 2020-07-31)
* ad45753 (aktualisierungen, 2020-07-31)
* 632b61c (reset-skript, 2020-07-31)
* db13c67 (Skripts entfernt, 2020-07-31)
* 65d2077 (Anwendungsprogramme aktualisiert, 2020-07-31)
* 6f98ab9 (in Ordner verschoben, 2020-07-31)
* eb8ebd6 (Skripts zur Steuerung der Datenbank verschoben, 2020-07-31)
* 19f55fe (nicht mehr aktuell, 2020-07-31)
* c3bf315 (Anwendungsprogramme für CRUD erstellt, 2020-07-30)
* 7d52269 (Aktualisierung Zustandsbefuellung, 2020-07-29)
* b01c5e6 (Skript zur Dockersteuerung, 2020-07-28)
* c858573 (Skripte zur Datenbanksteuerung, 2020-07-28)
* e97a670 (datentypen angepasst, 2020-07-28)
* 1e0d073 (init/zustand1.sh und init/users.sh entfernt weil ueberfluessig, 2020-07-28)
* a8dda66 (init/startup.sh, init/sample/zustand1.sql erweitert, 2020-07-28)
* f1ab481 (.gitignore aktualisiert, 2020-07-28)
* 58f8f35 (pr. key von standort, lager und abteilung zu character(5) geaendert, 2020-07-27)
* 1601c97 (uniqe(bundesland_id, stadt) zu unique(plz, stadt_geaendert), 2020-07-27)
* 18ecf44 (bundesland_id veraendert, 2020-07-27)
* 8d4166a (SQL Skripte für Zustandsbefüllung begonnen, 2020-07-27)
* 84efc98 (Constraint bundesland_format verallgemeinert, 2020-07-27)
* 83facca (Sequenz-, Trigger-, Funktions- und Tabellenerstellung fuer lagerort hinzugefuegt, 2020-07-27)
* e89adc6 (Schemaerstellung repariert Automatische Verbindung zu Projektdatenbank eingestellt, 2020-07-27)
* 15c51e5 (Nutzererstellung verbessert Schemaerstellung hinzugefuegt (zustand_1, zustand_2, zustand_3) Tabellendefinitionen fuer Zustaende hinzugefuegt, 2020-07
* 78d11a9 (users.sh in startup.sh umbenannt, 2020-07-26)
* d80a9ce (.gitignore aktualisiert, 2020-07-26)
* 0b2efdd (Funktions-, Sequenz-, Trigger- und Tabellenerstellung in SQL-Skript verlegt, 2020-07-26)
* 8021169 (Tabellen-, Sequenz-, Trigger- und Funktionserstellung in SQL-Skript create.sql verlegt, 2020-07-26)
* 9baf165 (aktualisiert(), create, sequenz(fuer lieferant_id) und trigger(fuer automatische letzte Aktualisierung) fuer lieferant hinzugefuegt, 2020-07-26)
* 3a1fead (create fuer abteilungen hinzugefuegt, 2020-07-26)
* 72bb94f (create fuer standort und lager hinzugefuegt, 2020-07-26)
* 56ce28f (create fuer eclass hinzugefuegt, 2020-07-26)
* 1b21edd (create fuer prioriserung der beschaffbarkeiten hinzugefuegt, 2020-07-26)
* faa47aa (create table für land, bundesland, stadt erstellt, 2020-07-26)
* 4cc694f (lagerist und abteilungsleiter als Rollen hinzugefuegt, 2020-07-26)
*   218e979 (Merge branch 'master' of https://gitlab-bs.kube.informatik.uni-halle.de/ajtrs/dbp20-projekt, 2020-07-24)
|\
| * d378f2f (Update README.md, 2020-07-22)
| * f6c6617 (Update README.md, 2020-07-22)
* | 2b5a98a (docker-compose.yml und init/users.sh, 2020-07-24)
|/
* f15ec61 (users.sh veraendert, 2020-07-22)
* 321c444 (Init-Ornder und unsers.sh erstellt, 2020-07-22)
* 374a4ad (Adminer aus yml Datei entfernt, 2020-07-21)
* 895485e (.env und .env.example zur .gitignore hinzugefuegt, 2020-07-21)
* 97aca38 (.env und .env.example entfernt, 2020-07-21)
*   3514fb9 (Merge branch 'master' of https://gitlab-bs.kube.informatik.uni-halle.de/ajtrs/dbp20-projekt, 2020-07-21)
|\
| * c85ab40 (Update README.md, 2020-07-21)
* | 62324f4 (.env und .env.example hinzugefuegt, 2020-07-21)
* | d0fdd09 (docker-compose.yml hinzugefeugt, 2020-07-21)
|/
* 3c2312c (readme aktualisiert, 2020-07-21)
* fcd4d5f (.gitigniore erstellen, 2020-07-21)
* e26a576 (README.md aktualisiert, 2020-07-08)
* 7d0e798 (Add README.md, 2020-07-08)
```

Letzter zur Bewertung heranzuziehender Commit: **09d075a**