technische universität dortmund

IT in Produktion und Logistik

Fachwissenschaftliche Projektarbeit

Konzeption einer SQL-Datenbank mit beispielhafter Befüllung aus einem produktionstechnischen Umfeld zu Lehrzwecken

Von
Lasse Jurgeleit
Matrikel-Nr.: 185841

Betreuer:
Dr.-Ing. Dipl.-Inform. Anne Antonia Scheidler
M.Sc. Nils Heidemann

Abgabedatum:
06.04.2020

Technische Universität Dortmund

Fakultät Maschinenbau

IT in Produktion und Logistik

Inhaltsverzeichnis

1. Einleitung

Die Verarbeitung von Daten mit Hilfe von Rechenanlagen war bereits in den 1960er-Jahren weit verbreitet. Es kam zur Entwicklung von Datenbank-Management-Systemen (DBMS), welche den Zugriff auf die in den Datenbanken gespeicherten Daten verwalten (Matthiessen und Unterstein 2007, S. 21). Zur Speicherung der Daten wurden verschiedene Datenbankmodelle entwickelt, wobei das relationale Datenbankmodell den meisten heutigen Datenbanksystemen zu Grunde liegt (Saake et al. 2018, S. 105). Für die Abfrage und Verwaltung von Daten in relationalen Datenbanken hat sich die Structured Query Language (SQL) durchgesetzt. SQL ist zu der dominierenden Sprache in der Datenbankwelt geworden und wird „[…] heute von allen kommerziellen und frei verfügbaren relationalen Datenbanken unterstützt“ (Saake et al. 2018, S. 201). Das Gebiet der Datenbanken hat sich ab den 2010er Jahren dynamischer entwickelt als je zuvor (Saake et al. 2018) und muss durch die fortschreitende Digitalisierung immer komplexere Anforderungen handhaben. Eine dieser Herausforderungen ist die Verwaltung von großen Datenmengen, der sogenannten BigData-Anwendungen. Besonders die Entwicklung der relationalen Datenbanken unter Verwendung von SQL greift diese Herausforderungen auf (Saake et al. 2018, S. 12). Da die Digitalisierung alle Bereiche der Arbeitswelt beeinflusst, ist es sinnvoll Studierenden aus dem produktionstechnischen Bereich Grundkenntnisse über Datenbanken und die Anwendung von Datenbank-Management-Systemen zu vermitteln.

Das Ziel dieser fachwissenschaftlichen Projektarbeit ist, den Teilnehmern des Moduls „Informationsaustausch produzierender Unternehmen“ vom Fachgebiet für IT in Produktion und Logistik (ITPL) zu ermöglichen, die vermittelten Inhalte anwendungsbezogen zu vertiefen. Die Lehrveranstaltung vermittelt den Studierenden unter anderem die Grundlagen der relationalen Datenbanken und der Datenbanksprache SQL. Für die anwendungsbezogene Vertiefung der Inhalte soll eine relationale Datenbank, welche ein produktionstechnisches Umfeld abbildet, erstellt werden. Anhand dieser sollen die Studierenden die Anwendung, des aus der Lehrveranstaltung vermittelten Wissens, mit Hilfe von Abfrageszenarien erlernen können. Dieses Hauptziel wird durch die Erarbeitung von folgenden aufeinander aufbauenden Teilzielen erfüllt. Als Grundlage soll im Bereich der produktionstechnischen Datenmodelle und Datenbanken recherchiert werden. Auf den Rechercheergebnissen aufbauend soll dann eine relationale Datenbank auf Basis von MySQL und mit Hilfe von phpMyAdmin, als Möglichkeit zu Administration und Verwaltung der Datenbank, erstellt werden. Weiterhin soll die relationale Datenbank mit einem realitätsnahen Datensatz befüllt

werden. Auf Grundlage dieser Datenbank sollen Abfrageszenarien konzipiert werden, welche sich auf die Rechercheergebnisse stützen. Dabei müssen zur Bearbeitung der Abfrageszenarien SQL-Befehle verwendet werden und es sollen den einzelnen Szenarien unterschiedliche Schwierigkeitsgrade zuzuordnen sein. Ebenso müssen die SQL-Befehle, welche zur Erarbeitung der Szenarien benötigt werden, aus der Lehrveranstaltung bekannt sein.

Die vorliegende Arbeit gliedert sich in mehrere aufeinander aufbauende Teile. Zu Beginn werden wichtige Fachbegriffe erläutert, um eine begriffliche Grundlage für die weitere Recherche zu schaffen. Anschließend werden die Grundlagen des Gebiets der Datenbanken erläutert und relationale Datenbanken in diesem Gebiet eingeordnet. Mit Hilfe dieser Grundkenntnisse wird anschließend der Bezug zum produktionstechnischen Umfeld dargelegt. Daraufhin werden Möglichkeiten zur Administration von MySQL-Datenbanken aufgezeigt und Bezug auf die Webanwendung phpMyAdmin genommen, die für die Lehrveranstaltung eingesetzt werden soll. Aufbauend auf den zusammengetragenen theoretischen Inhalten wird anschließend eine relationale Datenbank auf MySQL-Basis unter Verwendung von phpMyAdmin erstellt, welche ein produktionstechnisches Umfeld abbildet. Hierfür wird zu Anfang das abzubildende Umfeld definiert und die Anforderungen an die Datenbank erarbeitet. Auf Basis der Anforderungen wird anschließend eine relationale Datenbank konzipiert. Der Lösungsansatz wird zuerst in einem Entity-Relationship-Modell (ERM) visualisiert, bevor die Datenbank anschließend mit Hilfe von phpMyAdmin implementiert wird. Damit Studierende die implementierte Datenbank zu Übungszwecken nutzen können, wird diese mit Beispieldaten befüllt und Abfrageszenarien konzipiert. Außerdem wird den Studierenden die Ergebniskontrolle und das effektive Erarbeiten der Abfrageszenarien ermöglicht. Abschließend folgen Ausblick und Zusammenfassung der Arbeit.

2. Begriffliche Grundlagen und Einordnung

Das Ziel dieses Kapitels ist, eine begriffliche Grundlage zu schaffen, auf der die im Rahmen dieser Projektarbeit zu erstellende Datenbank basiert. Dafür werden in Kapitel 2.1 Modelle und Konzepte von Datenbanken zusammengetragen. Ebenfalls folgt eine kurze Einordnung relationaler Datenbanken im produktionstechnischen Umfeld und Möglichkeiten zur Administration von SQL-Datenbanken.

2.1. Datenbanken - Modelle und Konzepte

Zu Beginn dieses Kapitels erfolgt in 2.1.1 eine Einordnung von Datenbankmodellen. Dabei steht das relationale Datenbankmodell im Vordergrund. Darauf aufbauend werden Möglichkeiten des Datenbankdesigns von relationalen Datenbanken dargestellt und abschließend wird in Kapitel 2.1.3 die Datenbanksprache SQL eingeführt.

2.1.1. Einordnung von Datenbankmodellen

Die Aufgabe einer Datenbank ist die Verwaltung beliebiger Daten. Das Verwalten von Daten beinhaltet „[…] das Eingeben von neuen Daten, das Löschen veralteter Daten sowie das Nachführen bestehender Daten“ (Steiner 2017, S. 5). Dabei umfasst eine Datenbank die gespeicherten Daten und ein Datenbank-Management-System (DBMS) (Sauer und Grieger 1998, S. 12). Das DBMS bildet den Kern der Datenbank und ermöglicht das Verwalten der gespeicherten Daten. Zur Visualisierung und näheren Erläuterung lässt sich anhand Abbildung 1 der Begriff des Datenbanksystems einführen.

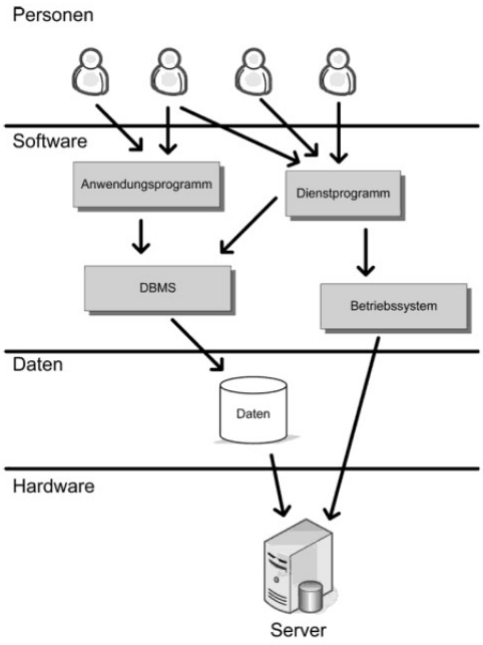

Abbildung 1: Das Datenbanksystem(Geisler 2014, S. 44)

Geisler (2014, S. 44) unterteilt dafür das Datenbanksystem in 4 Ebenen. Die Hardware-Ebene ist die Basis des Datenbanksystems. Unter der Hardware wird vom Clientrechner bis zum Datenbankserver jegliche Hardware, auf welche das Datenbanksystem aufgebaut ist, gezählt. Aus den Daten lassen sich die angefragten Informationen generieren und bilden somit den zentralen Punkt des Datenbanksystems. Jeglicher Zugriff auf die Daten erfolgt über die DBMS und bildet somit die Grundlage der Software-Ebene. Ergänzt wird die Software-Ebene durch weitere Programme, die je nach Personengruppe von Bedeutung sind. Unter Personen versteht man diejenigen, die mit dem Datenbanksystem arbeiten. Diese Personen werden nach Geisler (2014, S. 46) in funktionale Gruppen unterteilt, welche verschiedene Zugriffsrechte innehaben. Beispielhaft werden die Gruppen der reinen Anwender, der Datenbankdesigner und der Datenbankadministratoren genannt. Die reinen Anwender kommen meistens mit dem Anwendungsprogramm in Kontakt, erfassen Daten und werten diese aus. Die Datenbankdesigner hingegen legen die Strukturen der Datenbank fest und die Datenbankadministratoren sind dafür zuständig, dass das System zuverlässig und fehlerfrei läuft. Tätigkeiten des Administrators sind unter anderem das Anlegen und Zuordnen neuer Benutzer (Geisler 2014, S. 44–46).

Die Struktur der Informationen innerhalb einer Datenbank wird durch Datenbankmodelle festgelegt, welche jedoch keine Aussage über die Informationen selbst treffen (Saake et al. 2018, S. 67). Im Folgenden werden einige Datenmodelle beschrieben. Ausführlich wird auf das Entity Relationship-Modell (ERM) und das marktbeherrschende relationale Datenbankmodell eingegangen (Steiner 2017, S. 8). Weitere Modelle, wie das hierarchische, objektorientierte und objektrelationale Modell, werden

nur kurz behandelt. Auch wird auf die Datenbankmodelle eingegangen, welche unter dem Begriff NoSQL zusammengefasst werden und aufgrund des Bedarfs große Datenmengen hocheffizient zu verwalten entwickelt werden (Schicker 2017, S. 12).

Datenbankmodelle, welche unter dem Begriff NoSQL zusammengefasst werden, finden ihre Verwendung, wenn in kürzester Zeit auf Daten, welche in großen Datenmengen enthalten sind, zugegriffen werden soll. Beispielhaft dafür sind Anfragen auf Suchmaschinen. Um solch einen Zugriff zu gewährleisten, werden die Daten auf mehreren Servern performant verteilt und auf eine sekundengenaue Datenkonsistenz (Eindeutigkeit der Daten) verzichtet. Als Folge ist ein schneller Zugriff möglich, der jedoch nicht den aktuellen Stand der Daten garantiert. Beispiele für solche Datenbankmodelle sind dokumentenbasierte, spaltenorientierte oder graphenorientierte Datenbankmodelle (Schicker 2017, S. 15). Bereits in den 1960er Jahren wurden hierarchische Datenbanken eingesetzt. Der hierarchische Aufbau basiert auf einer Baumstruktur, welche eine geringe Redundanz, also geringe Mehrfachspeicherungen gleicher Informationen (Steiner 2017, S. 13), und kurze Zugriffszeiten gewährleistet. Durch die hohe Inflexibilität bei Änderungen der Datenstruktur, genügt sie allerdings den heutigen Bedarfen an Datenbanken nicht (Schicker 2017, S. 14). Bei objektorientierten Datenbanken wird das Paradigma der Objekt-Orientierung, welches z.B. in Programmiersprachen oder Betriebssystemen Anwendung findet, mit den Eigenschaften von Datenbanken kombiniert. Die vier wesentlichen Eigenschaften objektorientierter Datenmodelle sind die „[...] Modellierbarkeit komplexer Objekte, Unterstützung von Objekt-Identität, Unterscheidung von Typen und Klassen, Unterstützung von Klassenhierarchien" (Vossen und Lausen 2018, S. 27). Dabei steht das Objekt im Mittelpunkt, welches sowohl die Daten, als auch die für die Manipulation dieser Daten verwendeten Methoden, umfasst (Steiner 2017, S. 8). Zu beachten ist, dass objektorientierte Datenmodelle, im Rahmen der Objekt-Orientierung, auf verschiedene Weise entworfen sein können und somit die objektorientierten Datenbanken auf unterschiedlichen Datenmodellen basieren können (Vossen und Lausen 2018, S. 41–42). Zwischen zuvor beschriebenen objektorientierten und nachfolgend erläuterten relationalen Datenbanken hat sich eine Mischform entwickelt. Die sogenannten objektrelationalen Datenbanken basieren auf den relationalen Datenbanken, die durch Ansätze der Objektorientierung erweitert wurden. Dabei werden nicht alle Ideen der Objektorientierung ausgeschöpft (Schicker 2017, S. 13) und die objektrelationalen Datenbanken sind zu den relationalen Datenbanken kompatibel. (Steiner 2017, S. 8).

In relationalen Datenbanken werden die zu verwaltenden Daten in Tabellen gespeichert. Diese werden auch Relationen genannt und sind nach Themenkreisen geordnet (Steiner 2017, S. 10). Die physische und logische Datenhaltung ist dabei getrennt (Schicker 2017) und der Zugriff auf die gespeicherten Daten erfolgt stets über die Tabellen (Schicker 2017, S. 12). Grundlegende Arbeiten auf dem Gebiet des systematischen Aufbaus von Tabellen wurden von E. F. Codd Anfang der 1970er Jahre veröffentlicht und führten zu mathematisch fundierten Theorien, welche in der Struktur relationaler Datenbanken Anwendung finden (Schicker 2017, S. 25).

Formale relationale Bezeichner	Informelle Bezeichnung
Relation	Tabelle
Tupel	eine Zeile einer Tabelle
Kardinalität	Anzahl der Zeilen einer Tabelle
Attribut	eine Spalte einer Tabelle
Grad	Anzahl der Spalten einer Tabelle
Primärschlüssel	eindeutiger Bezeichner
Definitionsmenge	Menge aller möglichen Werte

Abbildung 2: Begriffe in relationalen Datenbanken nach (Schicker 2017, S. 26)

Die Struktur von eben erwähnten Relationen wird im Folgenden anhand von wichtigen Begriffen, die in Abbildung 2 zusammengestellt sind, und einer Beispielrelation erläutert. In Abbildung 3 ist die Relation *Student* zu sehen, welche Informationen von Studenten beinhaltet. Unterteilt werden kann eine Relation in den Kopf, mit den Attributen, und den Rumpf, mit den Tupeln. In dem Kopf der Relationen sind als Spaltennamen die Attribute aufgeführt. In der Beispielrelation sind das *Matrikel-Nr., Nachname,* usw.. Die Attribute korrespondieren jeweils zu einer Definitionsmenge und definieren die Menge, aus der die Tupeleinträge, auch Attributwerte genannt, stammen. Es gilt, dass jeder Tupeleintrag ein Element aus der Definitionsmenge des zugehörigen Attributs ist (Schicker 2017, S. 27). Weiterhin lassen sich die Begriffe Grad und Kardinalität einer Relation einführen. Der Grad einer Relation ist die Anzahl an Attributen, bzw. die Anzahl an Spalten. Zu beachten ist, dass der Grad beim Erstellen einer Relation festgelegt wird und nicht verändert werden kann. Bei dem Hinzufügen oder Weglassen von Attributen entsteht dementsprechend eine neue Relation, die wieder einen festgelegten Grad besitzt. Anders verhält es sich bei der Kardinalität. Die Kardinalität ist die Anzahl an Tupeln, bzw. die Anzahl an Zeilen. Durch das Hinzufügen oder Löschen von Tupeln ändert sich dementsprechend die Kardinalität einer Relation (Schicker 2017, S. 27). An der Beispielrelation *Student* würde sich die Kardinalität ändern, wenn sich beispielsweise ein Student immatrikuliert. Die Relation bekommt einen neuen Tupeleintrag und die Kardinalität steigt um Eins an. Um Tupel eindeutig voneinander unterscheiden zu können, werden Primärschlüssel verwendet. Der Primärschlüssel ist ein Attribut oder ein Zusammenschluss von Attributen mit bestimmten

Eigenschaften. In der Relation *Student* ist der Primärschlüssel die *Matrikel-Nr.*. Zwingend notwendig ist die Eindeutigkeit des Primärschlüssels. Durch ihn muss jeder Tupel eindeutig identifizierbar sein (Schicker 2017, S. 25–27). Dabei kann eine Relation mehrere Attribute besitzen, welche jeden Tubel eindeutig beschreiben. Dann wird ein Attribut als Primärschlüssel ausgewählt, die anderen sind Schlüsselkandidaten. Ein Primärschlüssel muss aber nicht nur aus einem Attribut bestehen, sondern kann aus mehreren Attributen zusammengesetzt sein (Schicker 2017, S. 32). Eine Übersicht über die erläuterten formalen Begriffe mit ihren informellen Bezeichnungen sind der Abbildung 2 zu entnehmen und in Abbildung 3 sind die formalen Begriffe graphisch, anhand der Beispielrelation *Student*, dargestellt.

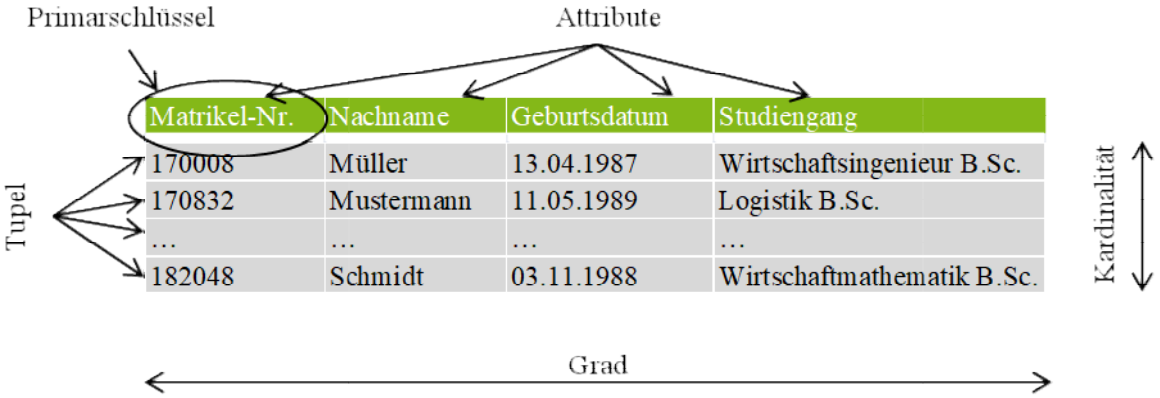

Abbildung 3: Begriffe anhand Beispielrelation „Student“ nach (Schicker 2017, S. 26); (Rabe 2018, S. 22)

Des Weiteren kann zur praktischen Anwendung der Begriff Relation, anhand von Schicker, spezifiziert werden. Dauerhaft gespeicherte Relationen, welche real existieren und einen festen Bestandteil der Datenbank bilden, werden auch Basisrelationen genannt. Von den Basisrelationen werden drei weitere Relationen abgeleitet. Die *Sichten*, auch Views genannt, sind Relationen, welche nur zu Teilen auf die Basisrelationen zugreifen. So können den Benutzern bestimmte Daten zur Verfügung gestellt, beziehungsweise verwehrt werden. Sie werden für den Schutz der Zugriffe genutzt. Ebenso von den Basisrelationen abgeleitet, sind die Abfrageergebnisse. Sie werden in Tabellenform ausgegeben und sind im Gegensatz zu den Sichten nur temporär im Arbeitsspeicher existent. Ähnlich zu den Abfrageergebnissen, sind die temporären Relationen. Diese sind Zwischenergebnisse, die zeitlich begrenzt, meist für eine Datenbanksitzung, gespeichert werden. Sie finden bei komplexen Zugriffen Anwendung, wenn das Abfrageergebnis durch mehrere aufeinander aufbauende Einzelschritte erreicht wird (Schicker 2017, S. 30).

Mit dem Wissen, welche Eigenschaften eine Relation besitzt, stellt sich die Frage, wie sich damit eine relationale Datenbank definieren lässt. "Eine relationale Datenbank ist [...] eine Ansammlung von Relationen, die zueinander in Beziehung stehen, und die in einem Verwaltungssystem verwaltet werden" (Schicker 2017, S. 29). Um Relationen in Beziehung zu setzen werden Fremdschlüssel

verwendet. Ein Fremdschlüssel ist ein Attribut oder mehrere Attribute, die sich auf einen Primärschlüssel einer in Zusammenhang stehenden Relation beziehen. Das bedeutet, dass die Anzahl der Attribute und die Definitionsgebiete des Fremdschlüssels der des Primärschlüssels entsprechen. Weiterhin kann ein Fremdschlüssel, im Gegensatz zum Primärschlüssel, Null-Werte besitzen. Doch ist es unzulässig, dass der Fremdschlüssel Werte annimmt, welche der Primärschlüssel, auf den sich bezogen wird, nicht enthält. (Schicker 2017, S. 38). Näheres zu Beziehungen zwischen Relationen bei der Datenbankentwicklung in 2.1.2.

2.1.2. Datenbankdesign mit dem Normalisierungsprozess und dem Entity-Relationship-Modell

Das Vorgehen beim Design einer Datenbank kann mit Hilfe des ERM erfolgen. Dabei werden im ersten Schritt alle eindeutig bestimmbaren Objekte, als Entitäten mit ihren Eigenschaften bestimmt und die Beziehungen zwischen den Entitäten ermittelt. Für die Überführung von dem ERM in die Datenbank werden die Entitäten in Relationen und die Eigenschaften in Attribute überführt. Relationen, die dann noch nicht in der dritten Normalform sind, werden in diese überführt. Die Beziehungen im ERM werden in Fremdschlüssel überführt und falls notwendig zusätzlich in Beziehungsrelationen (Schicker 2017, S.95). Neben der Verwendung des ERM, kann von dem Normalisierungsprozess Gebrauch gemacht werden. Durch ihn lassen sich Tabellen in normalisierte Relationen überführen. Dabei wird die Tabelle in mehrere Normalformen, mit steigenden Anforderungen, überführt. Nachfolgend werden die ersten drei Normalformen betrachtet. Höhere Normalformen werden hier nicht behandelt, da sie nicht von praktischer Bedeutung sind (Schicker 2017, S. 56).

Für die Erläuterung des Normalisierungsprozesses werden die funktionale und die volle Abhängigkeit eingeführt. Die Abhängigkeiten beziehen sich immer auf die Attribute und können aus mathematischer Sicht oder praktischer Sicht definiert werden. Hier werden nur die Definitionen aus der praktischen Sicht aufgeführt. Zu bemerken ist, dass die Definitionen beider Arten nicht blind für den Normalisierungsprozess angewendet werden können. Entscheidend sind die Erfahrung und die Kenntnis über das Umfeld des Anwenders.

Aus praktischer Sicht ist das Attribut oder die Attributkombination B funktional abhängig von dem Attribut oder der Attributkombination A, wenn B kein Teil von A ist, aber in derselben Relation enthalten ist und A der Primärschlüssel der Relation ist. Also wenn durch A eindeutig auf B geschlossen werden kann. Für die volle Abhängigkeit gilt: Das Attribut oder die Attributkombination

B ist dann von der Attributkombination A voll abhängig, wenn B von der Attributkombination, nicht aber von einem Teil der Kombination funktional abhängt (Steiner 2017, S. 53-54).

Mithilfe der funktionalen und vollen Abhängigkeit lassen sich nun die Anforderungen an die ersten drei Normalformen erläutern. Die erste Normalform ist erreicht, sobald die Relation ausschließlich atomare Attributwerte oder Nullwerte besitzt. Die zweite Normalform betrifft nur die Relationen, die einen zusammengesetzten Primärschlüssel aus mindestens zwei Attributen besitzen. Relationen mit einem Attribut als Primärschlüssel können aus der ersten direkt in die dritte Normalform überführt werden. Grundvoraussetzung für die Überführung in die zweite Normalform ist, wie in der Abbildung 4 dargestellt, dass die Relation bereits in der ersten Normalform ist. Außerdem muss jedes Attribut der Relation voll von dem zusammengesetzten Primärschlüssel abhängig sein. Es darf also kein Attribut geben, welches bereits durch einen Teil des Primärschlüssels eindeutig identifizierbar ist.

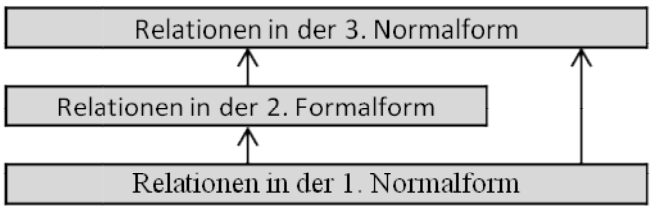

Abbildung 4: Die Normalisierungsebenen nach (Steiner 2017, S. 62)

Für die dritte Normalform muss die Relation, bei einfachem Primärschlüssel, bereits in der ersten Normalform oder, bei einem zusammengesetzten Primärschlüssel, in der zweiten Normalform, sein. Weiterhin verlangt die dritte Normalform, dass Attribute einer Relation nur zum Primärschlüssel funktional abhängig sein dürfen. Untereinander dürfen sie keine funktionale Abhängigkeit aufweisen. Ist dies erfüllt, befindet sich die Relation in der dritten Normalform und kann als normalisierte Relation bezeichnet werden (Schicker 2017, S. 56).

Für die Übertragung der realen Welt in eine Datenbank, mit Hilfe des ERM, werden Entitäten, Eigenschaften und Beziehungen verwendet. Reale eindeutig identifizierbare Objekte jeglicher Form werden im ERM als Entitäten übernommen und können als Rechtecke dargestellt werden. Eigenschaften können als Ellipsen dargestellt und den Entitäten zugeordnet werden. Beziehungen zwischen den Entitäten werden durch einen Beziehungstext und mit Hilfe der Beziehungswerte c, 1 und m dargestellt (Schicker 2017, S. 78-80). Beispielhaft ist in Abbildung 5 ein ERM abgebildet, welches die Entitäten *Person* und *Abteilung* abbildet. In den Ellipsen sind die Eigenschaften an der jeweiligen Entität angeordnet. Im vorliegenden Beispiel von Personen und Abteilungen einer Firma, könnten *Gehalt, Personalnummer* und *Name,* Eigenschaften für die Entität *Person* und *Budget, Abteilungsnummer* und das *Gebäude,* Eigenschaften für die Entität *Abteilung* sein. Drückt man die

Beziehung wörtlich aus, so enthält eine Abteilung mehrere Personen. Im vorliegenden Beispiel wird davon ausgegangen, dass eine Abteilung mehrere Personen beschäftigt, eine Person aber nur bei genau einer Abteilung beschäftigt sein kann. Der Annahme entsprechend liegt eine m zu 1 Beziehung vor.

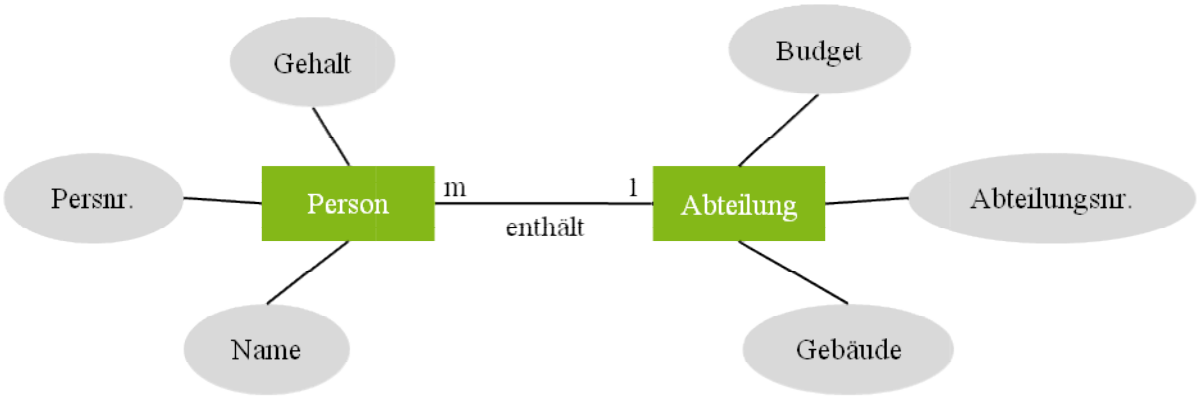

Abbildung 5: Beispiel Chen Notation nach (Schicker 2017, S. 79)

Wie den bereits erwähnten Beziehungswerten zu entnehmen, ist dies nicht die einzig mögliche Beziehung. Zur Erläuterung der Beziehungen werden im Folgenden die Definitionsbereiche der Beziehungswerte und alle Kombinationsmöglichkeiten dargelegt.

Der Beziehungswert c beinhaltet Null und Eins, während m bzw. n Werte größer gleich Null annehmen können. Werden alle Kombinationsmöglichkeiten der Werte betrachtet, erhält man eine Beziehungsmatrix, wie in Abbildung 6 dargestellt. Aus Gründen der Symmetrie können die oberen drei Felder unbeachtet bleiben und auch die 1 zu 1 Beziehung wird nicht weiter betrachtet, da diese in der praktischen Anwendung relationaler Datenbanken nicht vorkommt. Dies ist dadurch bedingt, dass nicht zur selben Zeit auf zwei Relationen zugegriffen werden kann. So gibt es immer einen Zeitpunkt, an dem der Eintrag einer potenziellen 1 zu 1 Beziehung, nur in einer Relation vorhanden ist (Schicker 2017, S. 82-83).

Beziehungen	c [0;1]	1	m
c [0;1]	A c c B	A c 1 B	A c m B
1	A 1 c B	A 1 1 B	A 1 m B
m	A m c B	A m 1 B	A m m B

Abbildung 6: ERM Beziehungsmatrix nach (Schicker 2017, S. 83)

Eine 1 zu c Beziehung ist eine relativ seltene Beziehung, in der ein Eintrag der Relation A keinmal oder einmal in der Relation B vorkommt. Andersherum muss jeder Eintrag der Relation B in der Relation A vorkommen. Eine Beziehung bei denen kein, ein oder mehrere Einträge einer Relation auf genau einen Eintrag der in Beziehung stehenden Relation verweisen, nennt man m zu 1 Beziehung. So auch die Beziehung aus dem Beispiel in Abbildung 5. Entweder sind keine, eine oder mehrere Personen in einer Abteilung beschäftigt und eine Person muss in genau einer Abteilung beschäftigt sein. Kommt es bei einer anderen Gegebenheit jedoch vor, dass Personen nicht zwingend in einer Abteilung beschäftigt sein müssen, so ist die Beziehung, getreu der Beziehungswerte, eine m zu c Beziehung. Als Spezialfall der m zu c Beziehung gilt die c zu c Beziehungen, welche nur selten Anwendung findet. Die letztmögliche Beziehung ist die m zu n Beziehung. Bei m zu n Beziehungen können mehrere Einträge einer Relation A auf denselben Eintrag in Relation B verweisen. Und auch umgekehrt können mehrere Einträge der Relation B auf denselben Eintrag in Relation A verweisen.

Anhand des vorangegangenen Beispiels der möglichen Beziehungen zwischen den Entitäten *Person* und *Abteilung* ist zu erkennen, dass ein exaktes Wissen des Umgebungskontextes, für die korrekte Bestimmung einer Beziehung, von Nöten ist. Daher ist es bereits im Datenbankdesign notwendig, Besonderheiten der Umgebung zu beachten (Schicker 2017, S. 83-84).

Beziehung	Überführung in Relation und Fremdschlüssel
m zu n	Erfordert Beziehungsrelation mit zwei m zu 1 oder m zu c Beziehungen
m zu c	Hinzufügen eines Fremdschlüssels zur m Relation
m zu 1	wie bei m zu c und Fremdschlüssel ist NOT NULL
c zu c	wie bei m zu c und Fremdschlüssel ist UNIQUE
c zu 1	wie bei m zu c und Fremdschlüssel ist NOT NULL und Fremdschlüssel ist meist Primärschlüssel

Abbildung 7: Überführung der Beziehungen in Relationen und Fremdschlüssel nach (Schicker 2017, S. 90)

Um die Beziehungen, welche im ERM visuell dargestellt werden, in relationalen Datenbanken darzustellen, müssen die Beziehungen aus dem ERM in Fremdschlüssel und Beziehungsrelationen überführt werden. Die Art der Überführung hängt dabei von der Art der Beziehung ab. Eine Übersicht der verschiedenen Überführungen kann der Abbildung 7 entnommen werden. Die Überführung einer m zu n Beziehung erfordert eine zusätzliche Relation. Durch diese sogenannte Beziehungsrelation entstehen zwei m zu 1 oder m zu c Beziehungen und die zusätzliche Relation enthält zwei Fremdschlüssel. Beide verweisen auf die Primärschlüssel der jeweilig beteiligten Entitäten. Zusätzlich bilden die Fremdschlüssel zusammen einen Schlüsselkandidaten der Beziehungsrelation. Die m zu 1 Beziehungen können durch das Einfügen eines Fremdschlüssels in

die m Beziehung erfolgen, wobei zu beachten ist, dass der Fremdschlüssel keine Nullwerte annehmen darf. Grundsätzlich gleich verhält es sich bei m zu c Beziehungen. Die Beziehungen unterscheiden sich jedoch dadurch, dass der Fremdschlüssel der m zu c Beziehung durchaus Nullwerte annehmen darf. 1 zu c Beziehungen werden durch einen Fremdschlüssel in der c Relation überführt. Der Fremdschlüssel darf keine Nullwerte besitzen und ist gleichzeitig ein Schlüsselkandidat. Im seltenen Fall einer c zu c Beziehung kann der Datenbankdesigner selbst entscheiden in welche Relation er den Fremdschlüssel einfügt, wobei es sich anbietet die logischen Zusammenhänge der Relationen bei der Entscheidung zu beachten. Weiterhin darf der Fremdschlüssel Nullwerte annehmen, muss aber eindeutig sein (Schicker 2017, S. 86-90).

2.1.3. Die Datenbanksprache SQL

Die Datenbanksprache SQL "ermöglicht dem Benutzer die Kommunikation mit dem eigentlichen Datenbanksystem und das Abfragen, Manipulieren und Schützen (Datenschutz) von Daten" (Steiner 2017, S.141). Zunächst wurde SQL als Sprache direkt vom Endbenutzer verwendet. Jedoch werden inzwischen, unter Verwendung von SQL, graphische Oberflächen, welche die Schnittstelle zum Anwender bilden, benutzt. Dafür erfolgt durch den Datenbankhersteller die SQL-Implementierung. Jedoch ist anzumerken, dass diese Implementierungen geringe Abweichungen besitzen. Dies führt dazu, dass für die Nutzung der Datenbanken von den einzelnen Herstellern geringe Anpassungen an der Syntax notwendig sind (Schicker 2017, S. 99-100). In Hinblick darauf, wird bei der folgenden Zusammenstellung ausgewählter SQL-Befehle, die Syntax, auf die in der Projektarbeit benutzen Oberfläche, angepasst. Da im Laufe der Projektarbeit phpMyAdmin verwendet wird, wird die hier verwendete Syntax mit phpMyAdmin kompatibel sein. Weiterhin werden im Folgenden nur die SQL-Befehle erläutert, die in dieser Projektarbeit angewendet werden und bilden nicht den gesamten Umfang von SQL ab.

Die Datenbanksprache SQL lässt sich in drei Untersprachen unterteilen, denen unterschiedliche Befehle zugehören. Mit der Data Definition Language (DDL) kann die Datenbank erstellt, geändert oder gelöscht werden. Veränderungen an den Daten der Datenbank können durch die Befehle der Data Manipulation Language (DML) erfolgen und mit Hilfe der Data Control Language (DCL) können die Daten geschützt und Zugriffsrechte verteilt werden (Geisler 2014, S. 210–216). In der Vorlesung wird vorwiegend die DML und zusätzlich die DDL behandelt. In Abbildung 8 sind Befehle, die in der Lehrveranstaltung Informationsaustausch produzierender Unternehmen (Rabe 2019) vorwiegend vorkommen, zusammengestellt und anhand Steiners (Steiner 2017, S 142-165) Ausführungen erläutert.

Sprache	Befehl		Beschreibung	Syntax
DDL	CREATE TABLE		Erzeugung von Basisrelationen	CREATE TABLE Tabellenname (Attributname_1 Datentyp [Einschränkung], Attributname_2 Datentyp [Einschränkung], Attributname_n Datentyp [Einschränkung]);
DML	INSERT		Einfügen neuer Tupel	INSERT INTO Tabellenname (Attribut1, Attribut2...) VALUES (Attributwert1, Attributwert2, ...);
	UPDATE		Nachführen bestehender Datensätze	UPDATE Tabellenname SET Attribut1=Ausdruck1, Attribut2=Ausdruck2, ... [WHERE Bedingung für Update];
	DELETE		Löschen von Datensätzen	DELETE FROM Tabellenname [WHERE Bedingung für Delete];
	SELECT		Abfrage von Datensätzen	SELECT Attributname_1, Attributname_2, ... FROM Tabellenname;
		DISTINCT	identische Datensätze werden nur einmal angezeigt	SELECT DISTINCT Attributenliste FROM Tabellenname;
		ORDER BY	Ordnen der anzuzeigenden Datensätze	SELECT Attributliste FROM Tabellenname ORDER BY Attributname [DESC oder ASC];
		JOIN	Verknüpfung von Relationen	SELECT Attributliste FROM Haupttabelle [join-typ] JOIN verknüpfteTabelle On Bedingung;
		COUNT	Zählen von Datensätzen	SELECT COUNT(Attributname1)Attributname2 FROM Tabellenname;
		SUM	Addiert Werte numerischer Attribute	SELECT SUM(Attributname) FROM Tabellenname;

Abbildung 8: Übersicht ausgewählter SQL-Befehle (eigene Darstellung)

Den Spalten auf der linken Seite kann die Zuordnung der Befehle zu der DDL und DML entnommen werden. So gehört der Befehl *CREATE TABLE* zu der DDL und *INSERT; UPDATE; DELETE* und *SELECT* zu der DML. Außerdem sind dem Befehl *SELECT* weitere Befehle zugeordnet. Diese werden, wie der Beschreibung und Syntax zu entnehmen, zur Erweiterung des Select-Befehls verwendet. Für das Verständnis der in Abbildung 8 vorgestellten Syntax, ist anzumerken, dass die teilweise verwendeten eckigen Klammern, optionale Schlüsselwörter der SQL-Anweisung aufzeigen, die den SQL-Befehl ergänzen können.

Wie der Abbildung 8 zu entnehmen, kann durch den Befehl *CREATE TABLE* eine Basisrelation erzeugt werden. Dabei muss, wie der Syntax zu entnehmen, der Datentyp der Attribute festgelegt werden und es können Einschränkungen formuliert werden. So können beispielsweise durch NOT NULL leere Attributwerte verhindert werden (Steiner 2017, S. 143). Durch den Insert-Befehl können Datensätze einer Relation hinzugefügt werden. Bereits vorhandene Datensätze werden durch *UPDATE* aktualisiert. Dafür werden die zu aktualisierenden Attribute und der neue Wert festgelegt.

Dazu kann mit *WHERE* eine Bedingung gestellt werden, mit der die betroffenen Attributwerte festgelegt werden können. Ohne eine Bedingung werden alle Attributwerte geändert (Steiner 2017, S. 151). Das Löschen von Datensätzen erfolgt durch *DELETE.* Hier verhält es sich wie bei dem Nachführen von Datensätzen. Durch eine Bedingung können bestimmte Datensätze gelöscht werden. Ohne Bedingung werden alle vorhandenen Datensätze gelöscht (Steiner 2017, S.152). Datenabfragen werden immer mit *SELECT* eingeleitet und können, wie der Abbildung 8 zu entnehmen ist, um weitere Schlüsselwörter erweitert werden (Steiner 2017, S.152). Wird der Select-Befehl alleine verwendet, können alle Datensätze der angegebenen Attribute einer Tabelle ausgegeben werden. Durch *DISTINCT* werden gleiche Einträge nur einmal ausgegeben (Steiner 2017, S.152) und durch die Verwendung von *ORDER BY* kann die Ausgabe der Datensätze sortiert werden. Sortiert wird alphabetisch oder nach Nummern. Für eine absteigende Sortierung wird das Schlüsselwort *DESC* verwendet und *ASC* für eine aufsteigende Sortierung (Steiner 2017, S. 156-157). Mit dem Befehl *COUNT* kann die Anzahl an Einträgen eines Attributs ausgegeben werden. Der Attributname1 steht hier für das Attribut, von dem die Einträge gezählt werden sollen. Das Ergebnis wird unter dem Attributnamen2 angegeben (Steiner 2017, S. 154-155). Der Befehl *SUM* ermöglicht die Addition von nummerischen Attributwerden (Steiner 2017, S. 158). Des Weiteren können Relationen verknüpft werden. Dafür kann der INNER, LEFT oder RIGHT JOIN verwendet werden. Dabei ist darauf zu achten, dass die Attribute den Relationen zugeordnet werden. Dies ist durch die Syntax *Tabellenname.Attributname* möglich. Der INNER JOIN zeigt nur die Datensätze an, die der Bedingung entsprechen. Im Gegensatz dazu werden beim LEFT JOIN alle Datensätze der Haupttabelle angegeben und von der verknüpften Tabelle genau die Datensätze, welche die Bedingung erfüllen. Beim RIGHT JOIN ist dies genau umgekehrt, sodass alle Datensätze der verknüpften Tabelle ausgegeben werden (Steiner 2017, S.163-164). Abschließend ist anzumerken, dass die Kombination einiger Befehle möglich ist und spezielle Abfragen ermöglichen kann.

2.2. Relationale Datenbanken im produktionstechnischen Umfeld

Die Produktionstechnik umfasst die „Gesamtheit wirtschaftlicher, technologischer und organisatorischer Maßnahmen, Verfahren und Methoden für die Herstellung industrieller Güter und Dienstleistungen aller Art" (Skolaut 2014, S. 968). Unter der Produktionstechnik wird die Fertigungs-, Energie- und Verfahrenstechnik eingeordnet. Die Verfahrenstechnik umfasst chemisch-physikalische oder biologische Vorgänge, die für die Erzeugung eines Produktes verwendet werden. „Die Energietechnik hingegen dient zu Gewinnung, Umwandlung, Transport, Speicherung und

Nutzung von Energie in allen Formen" (Skolaut 2014, S. 968). Die Fertigungstechnik umfasst die „Lehre von der wirtschaftlichen Herstellung geformter Werkstücke" (Skolaut 2014, S.968). Diese Herstellung geometrisch bestimmter Festkörper wird durch die Fertigungstechnik als Fertigungsverfahren definiert (Skolaut 2014, S.968).

Ein produzierendes Unternehmen verwendet für die Herstellung unterschiedliche Ressourcen, welche diversen Kategorien zugeordnet werden können (Schuh und Kampker 2011, S162). Eine Ressource ist Wissen, welche in Kategorien, wie dem individuellen und kollektiven Wissen oder expliziten und impliziten Wissen eingeteilt werden kann. Explizites Wissen, welches standardisierbar und quantifizierbar ist, kann in Datenbanken angelegt werden (Schuh und Kampker 2011, S.168-S.169). Für die Nutzung von Wissen ist es von Bedeutung einen zielorientierten Zugang zu Informationen sicherzustellen, der eine Transformation in Wissen ermöglicht. Ein solches Wissensmanagement zieht sich dabei durch alle Hierarchiestufen entlang der Wertschöpfungskette (Schuh und Kampker 2011, S.169- S.170). Ein System, welches Informationen verwaltet, beschafft und bereitstellt, kann als Informationssystem bezeichnet werden (Schuh und Kampker 2011, S 177). Für ein Informationssystem muss ein Modell der Daten aufgestellt werden. Die Daten bilden dann die Basis des Informationssystems (Schuh und Kampker 2011, S.180). Die verschiedenen Datenmodelle wurden bereits in Kapitel 2.1.1 erläutert. Das marktbeherrschende Datenmodell für Datenbanken ist dabei das relationale Datenbankmodell (Steiner 2017, S. 8).

2.3. Möglichkeiten zur Administration von MySQL-Datenbanken

Zu Anfang wird erklärt, was MySQL-Datenbanken sind und wie diese sich entwickelt haben. Ebenso erfolgt eine kurze Einführung in die Entstehung der MariaDB. Aus den Erkenntnissen daraus kann dann nachvollzogen werden, welche Datenbank für diese Projektarbeit, unter Verwendung von phpmyAdmin, benutzt wird. Anschließend wird auf die Anwendung phphmyAdmin und das Programmpaket XAMPP eingegangen.

MySQL ist als Datenbankanbieter für die Hardware und die logische Datenhaltung verantwortlich (Schicker 2017, S.181) und "[…] ist eine leistungsfähige Datenbank" (Schicker 2017, S.179). Eine für den Benutzer entwickelte graphische Oberfläche bieten dagegen Anwendungsanbieter an (Schicker 2017, S.181). MySQL, welche seit 1994 entwickelt wird, wird seit 2010 von Oracle weiterentwickelt. Der Schwerpunkt liegt seitdem nicht, wie bei der vorherigen Entwicklung, auf den frei verfügbaren Datenbank-Servern, sondern auf den kommerziellen. Daraufhin hat der Mitinitiator von MySQL Michael Widenius begonnen die MariaDB zu entwickeln. Hier liegt der Schwerpunkt

auf den open-source Lizenzen (Michael Rüttger 2019). Es ist dabei anzumerken, dass sich MariaDB und MySQL nur durch interne Umsetzungen unterscheiden (Michael Rüttger 2019).

Die Administration einer MySQL Datenbank kann, wie der Internetseite von phpMyAdmin entnommen werden kann, von phpMyAdmin gehandhabt werden. Dabei unterstützt die Anwendung sowohl MySQL, als auch MariaDB und stellt eine graphische Oberfläche zur Verfügung (phpMyAdmin 2020). Somit kann phpMyAdmin als Anwendungsprogramm gesehen werden. Die Einordnung eines Anwendungsprogramms im Datenbanksystem wurde bereits in Kapitel 2.1, anhand der Abbildung 1, vorgenommen. Unter anderem kann phpMyAdmin mit Hilfe von XAMPP verwendet werden. XAMPP ist ein frei verfügbares Softwarepaket, mit dem ein lokaler Apache Server mit der MariaDB installiert und konfiguriert werden kann. So lassen sich Konfigurationsschritte sparen. Es ist jedoch anzumerken, dass dafür Einschränkungen in Hinblick auf die Sicherheit zugelassen werden (XAMPP 2020).

3. Erstellen einer MySQL-Datenbank zu Lehrzwecken

Aufbauend auf den Rechercheergebnissen wird in diesem Kapitel eine Datenbank, welche für Übungszwecke genutzt werden soll, erstellt. Dafür wird ein produktionstechnisches Umfeld definiert, welches in einem ERM dargestellt wird. Anschließend folgt die Überführung in Relationen, die mit Hilfe von phpMyAdmin implementiert werden. Abgeschlossen wird das Kapitel mit der Erzeugung eines Datensatzes, welcher in die erstellten Relationen importiert wird.

3.1. Konzeptionierung der relationalen Datenbank

Zu Beginn, der in Kapitel 3 angestrebten Erstellung einer Datenbank, wird eine relationale Datenbank konzeptioniert. Dafür wird in Kapitel 3.1.1 ein Umfeld definiert, welches in Kapitel 3.1.2 in einem ERM dargestellt wird. Damit soll die Grundlage für die weitere Erstellung der Datenbank gelegt werden.

3.1.1. Definition eines produktionstechnischen Umfelds

Für die Definition des abzubildenden Umfelds wird zu Beginn der grobe Aufbau festgelegt. Darauf aufbauend werden in mehreren Schritten die feineren Strukturen erarbeitet. So wird versucht ein möglichst detailliertes Umfeld, mit klaren Anforderungen an die Datenbank, zu definieren, um den folgenden Datenbankentwurf zu erleichtern.

Fundamental soll das Umfeld aus einer Fertigung bestehen. In dieser Fertigung werden aus einzelnen Komponenten, die von Zulieferern bezogen werden, verschiedene Endprodukte gefertigt. Der im hier definierten Umfeld zu betrachtende Materialfluss, beginnt beim Wareneingang, wo Teile der Zulieferer ankommen. Darauf folgt die Fertigung und schließlich werden die Endprodukte im Warenausgang abtransportiert. Für die Lieferungen der Zulieferer wird ein Eingangslager bereitgestellt. In dieses werden alle ankommenden Teile eingelagert, bevor sie in die Fertigung kommen. Des Weiteren werden die gefertigten Produkte in einem zentralen Ausgangslager zwischengelagert. Die Kapazitäten für den Warenausgang werden durch den Bestand des Ausgangslagers bereitgestellt. Für die weitere strukturelle Festlegung werden die Anzahl der gefertigten Produkte und die dafür benötigten Zuliefererprodukte festgelegt. Dabei ist der Zweck,

dem das Umfeld dienen soll, zu beachten. Das Umfeld soll in einer Datenbank abgebildet werden, welche Studierende zur Anwendung von SQL-Befehlen nutzen sollen. Daher wird bei der Festlegung der Produktpalette von komplexen Strukturen abgesehen. Damit soll die Einarbeitungszeit in die Strukturen des Umfeldes gering gehalten werden. Aus eben diesen Gründen soll die Produktpalette aus vier Produkten bestehen, welche aus jeweils vier Komponenten gefertigt werden. Weiterhin soll jede Komponente nur für ein Endprodukt verwendet werden können. Demnach gibt es vier Produkte, die in der Fertigung produziert werden und 16 Komponenten, die jeweils einem Produkt fest zugeordnet sind. Weiterhin wird zwischen einem Produkt und einem Teil unterschieden. So kann jedes produzierte Teil einzeln identifiziert werden. Ebenso ist jedes Teil einem Produkt zugeordnet. Diese Unterscheidung ist auf eigengefertigte Produkte und auf die Produkte der Zulieferer anzuwenden. Am Beispiel eines Produktes ausgedrückt, besteht ein Produkt aus vier Zuliefererprodukten. Und ein gefertigtes Teil besteht aus 4 Zuliefererteilen. Dabei ist das gefertigte Teil einem Produkt zugehörig und die Zuliefererteile den Zuliefererprodukten.

Außerdem müssen die Grenzen des Umfelds, welches in dieser Projektarbeit weiter verwendet wird, festgelegt werden. Aus denselben Gründen wie bereits bei der Wahl der Produktpalette wird darauf geachtet, den Umfang begrenzt zu halten. Daher werden genaue Informationen der Zulieferer und Kunden im Weiteren außer Acht gelassen. Aus Sicht des Umfeldes kommen lediglich Komponenten nicht weiter benannter Zulieferer im Eingangslager an und werden nach der Fertigung aus einem Ausgangslager an nicht weiter bekannte Kunden weitertransportiert.

Nun werden die internen Abläufe und Strukturen festgelegt. Grundlegend wird die Annahme getroffen, dass die Fertigung an einem einzigen Standort stattfindet. An diesem Standort werden die Produkte in Hallen gefertigt. In den Hallen wird immer nur ein Produkt zur selben Zeit gefertigt. Diese Annahmen folgen aus der bereits angedeuteten Überlegung der Vermeidung von komplexen Zusammenhängen und die Fokussierung auf den Verwendungszweck des Umfeldes. Darauf stützend werden weitere Annahmen für die Fertigung getroffen. So sollen die Produkte von Fertigungsrobotern produziert werden. Ein Roboter übernimmt die gesamte Fertigung eines Teils. So ist ein Roboter zumindest temporär an ein Teil und auch an ein Produkt gebunden, kann aber theoretisch alle Produkte fertigen. Bezogen auf die Produktpalette von vier Produkten folgt, dass es vier Hallen geben muss in denen produziert werden kann und mindestens ein Roboter in jeder Halle fertigen muss. Dies gilt, solange davon auszugehen ist, dass die Fertigung jedes Produkt gleichzeitig fertigt. Dies wird bis auf Weiteres angenommen.

Damit wurde das Umfeld soweit festgelegt, dass es in einem ERM dargestellt werden kann. Weitere Informationen über die Stückzahl, Tage an denen gefertigt wird oder Lagerkapazitäten, sind noch nicht festgelegt. Dies hat den Grund, dass der Umfang der Daten an dieser Stelle noch nicht festgelegt werden soll. Daher werden weitere Informationen, welche den Umfang an Daten beeinflussen, in Kapitel 3.3 festgelegt. Beim Erzeugen eines Datensatzes für Lernzwecke, soll dadurch noch Einfluss auf die Menge an Daten genommen werden können.

3.1.2. Visualisierung der Datenbank in einem ERM

Das zuvor definierte Umfeld wird nun in einem ERM dargestellt. Das ERM soll anschließend für die Überführung in eine Datenbank genutzt werden. Dabei wird die in Kapitel 2.1.2 erläuterte Überführung in ein ERM angewendet. Zu Beginn werden die Objekte, welche in dem zuvor definierten Umfeld vorkommen, als Entitäten übernommen und mit notwendigen Eigenschaften ergänzt. Anschließend werden die Beziehungen zwischen den Entitäten übernommen und die Beziehungswerte zugeordnet. Bei der Darstellung des ERM wird, im Gegensatz zu der Darstellung in Kapitel 2.1.2, auf die Darstellung der Eigenschaften in Form von Ellipsen verzichtet. Anstatt dessen werden die Eigenschaften unter der jeweiligen Entität aufgelistet, wie der Abbildung 9 zu entnehmen ist. Das hat den Grund, dass so Platz gespart werden und die Darstellung übersichtlicher sein soll.

Aus dem zuvor definierten Umfeld werden im Folgenden die Entitäten übernommen. Die in der Fertigung hergestellten Produkte werden unter der Entität *Produkt* übernommen. Da diese Produkte aus verschiedenen Produkten, welche von Zulieferern bezogen werden, bestehen, werden auch die *ZuliefererProdukte* als Entität übernommen. Weiterhin wird in dem Umfeld, wie in 3.1.1 zu entnehmen, zwischen Produkten und den einzelnen hergestellten Teilen unterschieden. Daher kommen die Entitäten *Teile* und *ZuliefererTeile* hinzu. Für den Wareneingang und Warenausgang wird ein *Eingangslager* und *Ausgangslager* verwendet, die ebenfalls als Entitäten übernommen werden. Da in dem definierten Umfeld die Kunden nicht näher beschrieben werden, treten sie auch nicht als Entität in dem ERM auf. Der Transport der gefertigten Produkte wird betrachten. Daher wird die Entität *Lieferung* übernommen. Dabei ist anzumerken, dass die Entität lediglich für die Lieferungen des Warenausgangs steht. Noch nicht übernommen sind die für die Fertigung benötigten Entitäten. Die Produkte werden von Fertigungsrobotern in verschiedenen Fertigungshallen produziert. Dementsprechend werden die Entitäten *Fertigungshalle* und *Fertigungsroboter* in das ERM aufgenommen.

Die, den Entitäten zugeordneten, Attribute können der Abbildung 9 entnommen werden. Unter jeder Entität sind die Attribute zeilenweise aufgelistet. Eine Eigenschaft, welche jede Entität in diesem Umfeld besitzt, ist eine eindeutige Identifikationsnummer. Die Produkte und Zuliefererprodukte

haben eine Artikelnummer. Demnach wird der Entität *ZuliefererProdukte* das Attribut *ArtikelNrZulieferer* und der Entität *Produkt* das Attribut *ArtikelNrProdukt* zugeordnet. Nach diesem Prinzip wird dem Eingangslager und Ausganslager jeweils eine Platznummer als Attribut zugeordnet. Ebenso wird den Teilen eine Teilenummer zugeordnet. Eine Nummer für die Zuliefererteile und eine für die produzierten Teile. Weiterhin wird der Entität *Fertigungshalle* eine Hallennummer und der Entität *Fertigungsroboter* eine Roboternummer zugeordnet. Schließlich wird der Entität *Lieferung* das Attribut *LieferungsNr* zugeordnet. Zusätzlich zu den bereits genannten Identifikationsnummern wird den Produkten, welche in der Fertigung produziert werden und welche von den Zulieferern stammen, ein Name zugeordnet. Für die Teile der Zulieferer und die Lieferungen wird das Datum als weiteres Attribut aufgenommen. So ist der Entität *ZuliefererTeile* das Attribut *Anlieferung* und der Entität *Lieferung* das *Lieferdatum* zugeordnet. Ebenso wird das Produktionsdatum, der in der Fertigung produzierten Teilen, der Entität *Teile* zugeordnet, wie der Abbildung 9 entnommen werden kann. Damit sind die Attribute, welche im Rahmen dieser Projektarbeit genutzt werden, den Entitäten zugeordnet und es können anschließend die Verbindungen der Entitäten dargestellt werden.

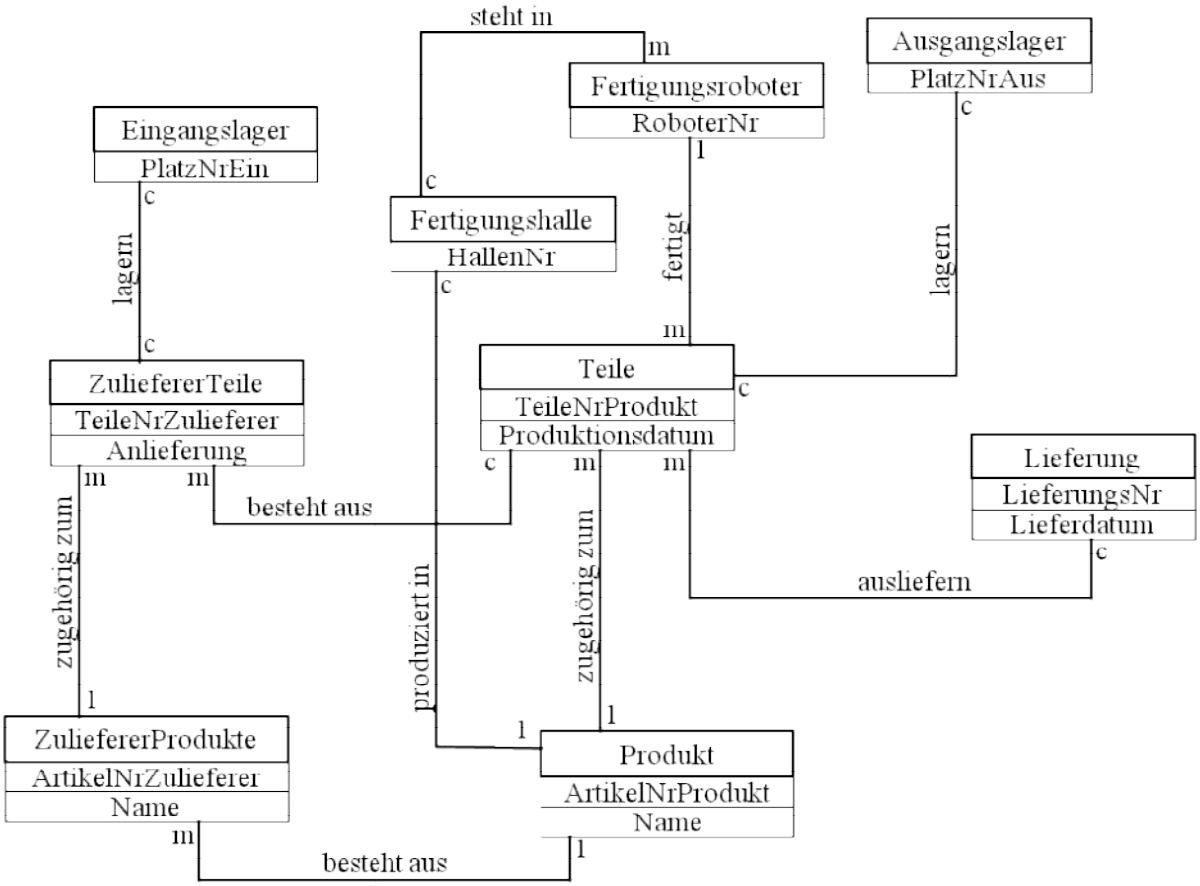

Abbildung 9 das ERM eines definierten Umfeldes (eigene Darstellung)

Zu Beginn werden die Beziehungen der Entität *Produkt* erläutert und daraufhin folgen alle weiteren Beziehungen. Wie in 3.1.1 beschrieben, besteht ein Produkt aus mehreren Zuliefererprodukten. Dementsprechend steht das Produkt mit den Zuliefererprodukten in Beziehung. Dabei kann ein Produkt mehrere Zuliefererprodukte enthalten und ein Zuliefererprodukt muss in genau einem Produkt enthalten sein. Daraus folgt, dass dem Produkt der Beziehungswert 1 zuordnet wird und den Zuliefererprodukten der Wert m. Zu beachten ist, dass damit die Annahme getroffen wird, dass es kein Zuliefererprodukt geben kann, welches nicht für ein Produkt verwendet wird und dass jedes Zuliefererprodukt nur für ein Produkt verwendet werden kann. Weiterhin sind dem Produkt gefertigte Teile zugehörig. Ein Teil gehört immer zu genau einem Produkt und von einem Produkt kann es mehrere produzierte Teile geben. Daher wird hier eine m zu 1 Beziehung gewählt, bei der das m der Entität *Teile* zugeordnet ist. Außerdem werden die Produkte in einer Fertigungshalle produziert, in der nur ein Produkt hergestellt wird. Weiterhin wird hier davon ausgegangen, dass es keine Fertigungshalle gibt, in der kein Produkt produziert wird oder welche nicht bereits für die Produktion eines Produktes vorbestimmt ist. Andersherum, wird die Annahme getroffen, dass es Produkte geben kann, welche keiner Halle zugeordnet sind. Das können beispielsweise Produkte sein, welche nicht mehr produziert werden oder Produkte, welche sich in einem Entwicklungsstatus befinden. Aus diesen Gründen wird zwischen den Entitäten *Produkt* und *Fertigungshalle* eine 1 zu c Beziehung gewählt. Des Weiteren steht die Fertigungshalle in Beziehung zu den Fertigungsrobotern, welche in den Fertigungshallen stehen. Ein Roboter kann zu einem Zeitpunkt nur in einer Halle stehen. Es könnte aber durchaus Roboter geben, die in keiner Halle stehen. Beispielsweise ein defekter Roboter der zur Reparatur in eine Werkstatt gebracht wurde. Es wird hier nicht weiter betrachtet, wie eine solche Reparatur ablaufen würde, aber für die Auswahl der Beziehungswerte wird die Annahme getroffen, dass es einen solchen Fall geben kann. Weiterhin können in einer Halle mehrere Roboter stehen. Aus diesen Gründen wird hier von einer m zu c Beziehung ausgegangen. Weiterhin wird die Beziehung zwischen den Fertigungsrobotern und den Teilen betrachtet. Wie in 3.1.1 festgelegt wird die gesamte Fertigung eines Teils von einem einzigen Roboter übernommen. Ein Roboter produziert in seinem Lebenszyklus jedoch mehrere Teile. Daher wird zwischen der Entität *Fertigungsroboter* und *Teile* eine 1 zu m Beziehung gewählt. Damit sind die Beziehungen der Entität *Teile* zu den Fertigungsrobotern und dem Produkt festgelegt. Weiterhin stehen die Teile zu dem Ausgangslager, der Lieferung und den Zulieferer Teilen in Beziehung. Nach der Produktion durch einen Fertigungsroboter werden die Teile in einem Ausgangslager zwischengelagert, bevor sie zum Kunden geliefert werden. Dabei liegen die Teile nur zeitlich begrenzt in dem Ausgangslager. Daraus folgt, dass ein Teil in dem Lager liegt oder aber nicht. Außerdem kann ein Teil zu einem Zeitpunkt nur einem Platz in dem Lager zugeordnet sein. Aus Sicht des Ausgangslagers kann ein Lagerplatz entweder belegt oder nicht belegt sein. Aber es kann keinen Lagerplatz geben, welcher zu einem

Zeitpunkt durch zwei Teile belegt ist. Aus den eben genannten Gründen wird sich zwischen den Teilen und dem Ausgangslager für eine c zu c Beziehung entschieden. Dies hat zu Folge, dass nur die aktuelle Lagersituation abgebildet wird. Es wird nicht betrachtet welchem Lagerplatz ein Teil zugeordnet war, bevor es zum Kunden ausgeliefert wurde. Es wird also angenommen, dass diese Information, in dem hier verwendeten Umfeld, keinen Nutzen hat. Nach der Zwischenlagerung in dem Ausgangslager werden die Teile als Teil einer Lieferung zu einem Kunden geliefert. Eine Lieferung beinhaltet mehrere Teile und ein Teil ist genau einer Lieferung, oder aber keiner Lieferung zugeordnet. So ist ein Teil, welches im Ausgangslager liegt noch keiner Lieferung zugeordnet. Daraus folgt eine m zu c Beziehung zwischen den Teilen und der Lieferung. Eine weitere Entität mit der die Teile in Beziehung stehen, sind die Zuliefererteile. Entsprechend dem in 3.1.1 definierten Umfeld besteht ein Teil aus mehreren Zuliefererteilen und ein Zuliefererteil ist in genau einem oder aber keinem Teil verbaut. Daraus folgt, die in Abbildung 9 dargestellte m zu c Beziehung zwischen den Zuliefererteilen und den gefertigten Teilen. Ein Beispiel für ein Zuliefererteil, welches in keinem Teil verbaut ist, sind die Zuliefererteile, die in dem Eingangslager liegen. So stehen die Zuliefererteile in Beziehung zu dem Eingangslager. Dabei verhält es sich wie bei dem zuvor beschriebenen Ausgangslager. Teile werden temporär genau einem Lagerplatz zugeordnet. Dementsprechend wird auch zwischen den Zuliefererteilen und dem Eingangslager eine c zu c Beziehung gewählt. Die letzte Beziehung in dem in Abbildung 9 dargestellten ERM, ist zwischen den Zuliefererteilen und den Zuliefererprodukten. Diese Beziehung verhält sich wie die Beziehung zwischen den Produkten und Teilen. Ein Zuliefererteil muss genau einem Zulieferprodukt zugeordnet sein und von einem Zulieferprodukt kann es mehrere Zuliefererteile geben. Das gesamte ERM kann der Abbildung 9 entnommen werden.

3.2. Implementierung der Datenbank in phpMyAdmin

Im Folgenden Abschnitt wird die in Kapitel 3.1 konzipierte Datenbank in phpMyAdmin implementiert. Damit wird die Grundlage gelegt, die Datenbank für Übungszwecke von SQL nutzen zu können. Begonnen wird mit der Einrichtung von phpMyAdmin mit Hilfe von XAMPP. Anschließend wird in Kapitel 3.2.2, das ERM aus Kapitel 3.1.2 in Relationen überführt und die Relationen werden schließlich mit Hilfe von SQL-Befehlen, welche in Kapitel 2.1.3 erläutert wurden, implementiert.

3.2.1. Einrichten von phpMyAdmin

Im Rahmen dieser Projektarbeit wird phpMyAdmin über die Software XAMPP verwendet. Dafür muss XAMPP heruntergeladen und eingerichtet werden. XAMPP wird hier über die Internetseite ***https://www.apachefriends.org/de/index.html*** heruntergeladen. Beim Installieren wird darauf geachtet, dass der Apache und MySQL Server und phpMyAdmin installiert wird. Daneben gibt es weitere Optionen, die installiert werden können, auf die hier jedoch nicht eingegangen wird. Nachdem XAMPP installiert ist, wird das Control Panel von XAMPP geöffnet. Über das Control Panel lässt sich nun das Modul *Apache* und das Modul *MySQL* starten. Daraufhin lässt sich phpMyAdmin in einem Browser öffnen. Dafür kann der Button *Admin*, in der Zeile des Moduls *MySQL*, gedrückt werden. Es kann aber auch ein beliebiger Browser und die Adresse *localhost/phpmyadmin/* geöffnet werden. Damit ist phpMyAdmin mit Hilfe von XAMPP eingerichtet und kann genutzt werden.

Abbildung 10 XAMPP Control Panel

Für die anschließende Implementierung, wird eine Datenbank angelegt, in welcher die Relationen implementiert werden können. Damit phpMyAdmin eine neue Datenbank anlegen kann, werden der Datenbankname und die Kollation festgelegt. Dementsprechend wir die Datenbank *produktionsdb* genannt und die Kollation *latin1_swedish_ci* gewählt. Somit ist die Datenbank erstellt. Durch XAMPP wird für die Datenbank ein Dateiordner angelegt. Der Ordner liegt unter dem Pfad xampp>mysql>data. Damit ist die Grundvoraussetzung für die Implementierung der Datenbank in phpMyAdmin geschaffen.

3.2.2. Implementierung des Datenbankmodells als MySQL-Datenbank

Vor der eigentlichen Implementierung werden die in dem ERM dargestellten Beziehungen überführt. Dafür wird die in 2.1.2 erläuterte Abbildung 7 hinzugezogen, mit welcher die Beziehungen durch Fremdschlüssel in Relationen überführt werden können. Außerdem werden die Fremdschlüssel je nach Beziehung eingeschränkt. Daraufhin werden den Attributen Datentypen zugeordnet, bevor die Relationen schließlich in phpMyAdmin implementiert werden. Ein Überblick über die überführten Relationen kann anhand der Tabelle in Abbildung 11 entnommen werden. Es werden die Relationen und alle, nach der Überführung existierenden, Attribute aufgezeigt. Dazu können die Datentypen und Einschränkungen, der Attribute, entnommen werden.

Zu Beginn wird jede existierende Beziehung betrachtet und überführt. Wie der Abbildung 7 entnommen werden kann, wird bei einer m zu 1 Beziehung der m Relation ein Fremdschlüssel zugeordnet, welcher nicht leer sein darf. Im Folgenden wird die Überführung der m zu 1 Beziehungen anhand der Beziehung zwischen der Relation *Produkt* und *ZuliefererProdukt* beschrieben. Alle weiteren m zu 1 Beziehungen werden nach dem gleich Vorgehen überführt, aber hier nicht im Einzelnen beschrieben. Der Abbildung 11 können jedoch die resultierenden Fremdschlüssel entnommen werden. Wie in 2.1.1 erläutert, bezieht sich ein Fremdschlüsselattribut auf den Primärschlüssel, der in Beziehung stehenden Relation. Für die Beziehung zwischen dem Produkt und den Zuliefererprodukten hat dies zu Folge, dass der Relation *ZuliefererProdukt* das Attribut *ArtikelNrProdukt* als Fremdschlüssel hinzugefügt wird. Zusätzlich wird die Bedingung gestellt, dass der Fremdschlüssel keine leeren Attributwerte annehmen darf. In Abbildung 11 ist diese Einschränkung durch den Zusatz *not null* gekennzeichnet. Nach diesem Prinzip werden alle weiteren m zu 1 Beziehungen überführt. In dem vorliegenden ERM sind dadurch zusätzlich die Beziehungen zwischen *Produkt* und *Teile*, *ZuliefererProdukt* und *ZuliefererTeile* und zwischen *Fertigungsroboter* und *Teile* betroffen. Durch die Überführung resultieren die Fremdschlüssel *ArtikelNrProdukt* und *RoboterNr* in der Relation *Teile.* In der Relation *ZuliefererTeile* kommt der Fremdschlüssel *ArtikelNrZulieferer* hinzu.

Weiterhin werden in dem vorliegenden ERM m zu c Beziehungen verwendet. M zu c Beziehungen werden nach der Abbildung 7 durch einen Fremdschlüssel, ohne weitere Einschränkungen, in der m Relation, überführt. Davon betroffen sind die Beziehungen zwischen *Teile* und *Lieferung*, *ZuliefererTeile* und *Teile* und zwischen *Fertigungsroboter* und *Fertigungshalle*. Durch die Überführung kommt der Fremdschlüssel *LieferungsNr* bei der Relation *Teile* hinzu. Weiterhin kommt bei der Relation *ZuliefererTeile* der Fremdschlüssel *TeileNrProdukt* und bei der Relation *Fertigungsroboter* der Fremdschlüssel *HallenNr* hinzu.

Die 1 zu c Beziehung zwischen den Relationen *Produkt* und *Fertigungshalle* wird entsprechend der Abbildung 7 durch einen Fremdschlüssel in der c Relation überführt. Dazu darf der Fremdschlüssel keine Nullwerte annehmen. Dementsprechend wird der Relation *Fertigungshalle* der Fremdschlüssel *ArtikelNrProdukt*, mit dem Zusatz *not null*, zugeordnet. Zuletzt gibt es eine c zu c Beziehung zwischen den Relationen *Teile* und *Ausgangslager* und zwischen *ZuliefererTeile* und *Eingangslager*. Die Überführung einer c zu c Beziehung erfolgt durch das Hinzufügen eines Fremdschlüssels, in einer der beiden Relationen und der Fremdschlüssel muss eindeutig sein. Für die Überführung muss also entschieden werden in welcher Relation ein Fremdschlüssel hinzugefügt wird. Für diese Entscheidung werden die logischen Zusammenhänge betrachtet. Dabei fällt auf, dass in beiden Fällen Teile in einem Lager gelagert werden. Lediglich die Art der Teile ist unterschiedlich. Zum einen sind es Zuliefererteile und zum anderen, die in der Fertigung produzierten Teile. Da sich die logischen Zusammenhänge dadurch nicht unterscheiden, wird die folgende Überlegung für die Entscheidung in beiden Fällen hinzugezogen. Um nicht zu jedem jemals produzierten oder angelieferten Teil eine Information zu dessen Lagerplatz zu speichern, wird sich für die Lager Relation entschieden. Somit können einige leere Einträge eingespart werden. Es werden die aktuell eingelagerten Teile auf die zumindest temporär festgesetzte Lagerkapazität aufgeteilt. Dementsprechend wird dem Eingangslager die Teilenummer der Zuliefererteile und dem Ausgangslager die Teilenummer der produzierten Teile, als Fremdschlüssel zugeordnet.

Für die Implementierung werden nun die Datentypen der Attribute festgelegt. Der zugeordnete Datentyp jedes Attributs kann der Spalte *Datentyp* in Abbildung 11 entnommen werden. Attribute, welche für die Speicherung von einem Datum genutzt werden, wird der Datentyp *date* zugeordnet. Im vorliegenden Fall sind davon die Attribute *Lieferdatum*, *Produktionsdatum* und *Anlieferung* betroffen. Dem Attribut *Name* von dem Produkt und den Zuliefererprodukten wird der Datentyp *varchar(200)* zugeordnet. Somit lassen sich Zeichenfolgen mit variabler Länge speichern, wobei eine maximale Anzahl von 200 Zeichen festgelegt ist. Alle weiteren Attribute sind Primärschlüssel, welche teilweise zusätzlich als Fremdschlüssel verwendet werden. Der Datentyp des Fremdschlüssels unterscheidet sich jedoch nicht von dem Datentyp des zugehörigen Primärschlüssels. Für die Primärschlüssel wird eine Zeichenfolge mit fester Anzahl an Zeichen ausgewählt. Dafür wird der Datentyp *char()* verwendet. Durch den Wert in der Klammer wird die Anzahl an Zeichen festgelegt. Wie der Abbildung 11 entnommen werden kann, unterscheiden sich die Längen der Zeichenfolgen je nach Attribut. Das hängt mit dem Aufbau der Identifikationsnummer zusammen, welcher in 3.3.2 zu jedem Primärschlüssel festgelegt wird.

Relation	Attribut	Datentyp	Einschränkung
Produkt			
	ArtikelNrProdukt	char(8)	Primärschlüssel
	Name	varchar(200)	
ZuliefererProdukte			
	ArtikelNrZulieferer	char(8)	Primärschlüssel
	Name	varchar(200)	
	ArtikelNrProdukt	char(8)	Fremdschlüssel, not null
Lieferung			
	LieferungsNr	char(8)	Primärschlüssel
	Lieferdatum	date	
Fertigungshalle			
	HallenNr	char(3)	Primärschlüssel
	ArtikelNrProdukt	char(8)	Fremdschlüssel, not null
Fertigungsroboter			
	RoboterNr	char(4)	Primärschlüssel
	HallenNr	char(3)	Fremdschlüssel
Teile			
	TeileNrProdukt	char(9)	Primärschlüssel
	Produktionsdatum	date	
	ArtikelNrProdukt	char(8)	Fremdschlüssel, not null
	RoboterNr	char(4)	Fremdschlüssel, not null
	LieferungsNr	char(8)	Fremdschlüssel
Ausgangslager			
	PlatzNrAus	char(8)	Primärschlüssel
	TeileNrProdukt	char(9)	Fremdschlüssel, not null
ZuliefererTeile			
	TeileNrZulieferer	char(9)	Primärschlüssel
	Anlieferung	date	
	ArtikelNrZulieferer	char(8)	Fremdschlüssel, not null
	TeileNrProdukt	char(9)	Fremdschlüssel
Eingangslager			
	PlatzNrEin	char(8)	Primärschlüssel
	TeileNrZulieferer	char(9)	Fremdschlüssel, unique

Abbildung 11 Überführung des ERM in Relationen (eigene Darstellung)

Mit Hilfe von den zuvor erarbeiteten und in Abbildung 11 zusammengetragenen Informationen lassen sich nun die Relationen in phpMyAdmin einfügen. Dafür kann in phpMyAdmin unter dem Reiter *SQL* der Quellcode ausgeführt werden. Die Implementierung erfolgt mit dem Befehl *CREATE TABLE*, unter Verwendung der in 2.1.3 beschriebenen Syntax. Da die, für die Implementierung benötigten, Informationen bereits der Abbildung 11 entnommen werden können, müssen diese Informationen lediglich in die Syntax gebracht werden. Die Syntax ist der Abbildung 8 zu entnehmen. Bei der Implementierung ist jedoch auf die Reihenfolge zu achten, in der die Relationen implementiert werden. Es ist nicht möglich ein Fremdschlüsselbezug herzustellen, ohne dass die in

Beziehung stehende Relation besteht. Das führt dazu, dass entweder die Fremdschlüsselbezüge festgelegt werden, wenn alle Relationen implementiert sind oder aber es wird eine Reihenfolge gefunden, bei der die Fremdschlüsselbezüge bereits festgelegt werden können. Eine mögliche Reihenfolge kann der Abbildung 11 entnommen werden. Die Implementierung erfolgt von oben nach unten. Beginnend mit der Relation *Produkt* und zuletzt wird die Relation Eingangslager implementiert. Bei der Reihenfolge wird beachtet, dass die Relation, auf die sich ein Fremdschlüssel bezieht, bereits existiert. Beispielsweise kann die Relation *ZuliefererProdukte* betrachtet werden. Damit das Attribut *ArtikelNrProdukt* als Fremdschlüssel festgelegt werden kann, muss die Relation *Produkt* mit dem Primärschlüssel *ArtikelNrProdukt* bereits existieren. Aus diesem Grund wird erst die Relation *Produkt* und anschließend die Relation *ZuliefererProdukte* implementiert. Nach diesem Kriterium, wird die Reihenfolge aller zu implementierenden Relationen festgelegt und kann wie bereits erwähnt anhand der Abbildung 11 abgelesen werden. Nach eben dieser Reihenfolge werden nun die Relationen implementiert. Der dafür verwendete Quellcode, richtet sich nach der, in Abbildung 8 dargestellten, Syntax und kann dem Anhang 1 entnommen werden.

3.3. Erzeugung eines Datensatzes für Lehrzwecke

Das Ziel dieses Kapitels ist, die in Kapitel 3.2 erstellte Datenbank mit Daten zu befüllen, sodass diese für Übungszwecke genutzt werden kann. Dafür werden in 3.1.1 Möglichkeiten der Datenbeschaffung betrachtet. Daran anschließend wird ein Datensatz erzeugt und in die bestehende Datenbank importiert.

3.3.1. Möglichkeiten der Datenbeschaffung

Zu Beginn werden zwei Möglichkeiten betrachtet, wie die in Kapitel 3.2 implementierte Datenbank mit Datensätzen befüllt werden kann. Dabei wird nicht nur die Verwendung einzelner SQL-Befehle, sondern auch die Möglichkeit des Importierens durch phpMyAdmin betrachtet. Wie bereits in Kapitel 2.1.3 erläutert wird für das Einfügen von neuen Tupeln der Befehl *INSERT INTO* verwendet. So lassen sich einzelne Tupel in eine bestimmte Relation einfügen. Wenn dieser Befehl manuell für jeden einzelnen Tupel geschrieben werden muss, kann es jedoch zu einem großen Zeitaufwand kommen. PhpMyAdmin bietet eine andere Möglichkeit. Mit der Funktion *Importieren* lassen sich Daten, welche in verschiedenen Dateiformaten gespeichert sein können, direkt in bestehende Relationen einfügen. Dabei wird erst eine Relation ausgewählt und dann das Fenster für das Importieren geöffnet. Eine Möglichkeit ist das Importieren von Excel-Dateien im CSV-Format.

In dem Fenster für das Importieren wird das Format *CSV using LOAD DATA* ausgewählt und die vorausgefüllten Einstellungen übernommen. Für das Importieren werden die Tupel in Tabellenform in der Excel-Datei gespeichert. Durch das Importieren werden dann die beschriebenen Zellen spaltenweise in die ausgewählte Relation eingefügt. Es ist nicht nötig die Spalten nach den Attributen zu benennen. Die Spalten müssen jedoch in der gleichen Reihenfolge angelegt sein, wie sie in phpMyAdmin angezeigt werden. Als Beispiel kann die Relation *Produkt* betrachtet werden. In phpMyAdmin wird als linkes Attribut die Artikelnnummer und als rechtes der Name angezeigt. Um nun die richtigen Datensätze einzufügen, müssen in der Excel-Datei die Attributwerte der Artikelnummern in der ersten Spalte gespeichert werden. Die Namen dementsprechend in der darauf folgenden Spalte.

Wie bereits am Ende des Kapitels 2.1.1 erläutert wurde, darf ein Fremdschlüssel keine Werte annehmen, welche der Primärschlüssel nicht besitzt. Für das Importieren einzelner Relationen hat dies zur Folge, dass die Datensätze, auf die sich ein Fremdschlüssel bezieht, bereits in der Datenbank eingefügt sein müssen. Aus ähnlichem Grund wurde bereits in Kapitel 3.2.2 auf die Reihenfolge beim Implementieren der Relationen geachtet. Es wurde eine Reihenfolge festgelegt, bei der eine Relation mit Fremdschlüssel erst dann implementiert wurde, wenn die in Verbindung stehende Relation bereits existiert. Diese Reihenfolge kann auch bei dem Importieren der Datensätze genutzt werden. Weiterhin muss beim Importieren von Nullwerten darauf geachtet werden, dass die Zellen in der Excel-Tabelle nicht leer sein dürfen, sondern mit *NULL* befüllt seien müssen. Ansonsten wird dem Attribut kein Nullwert hinzugefügt.

3.3.2. Erzeugung von Daten für die Datenbank

Für die Erzeugung der Daten wird das in Kapitel 3.1.1 definierte Umfeld hinzugezogen und weitere Annahmen getroffen. Durch weitere Annahmen wird versucht die Datenmenge logisch einzugrenzen, ohne Wiedersprüche innerhalb des Umfeldes zu erzeugen. Gespeichert werden die Daten in mehreren CSV-Dateien. Für jede Relation wird eine Datei angelegt. Die CSV-Dateien sollen, nachdem die Daten festgelegt sind, mit Hilfe von phpMyAdmin in die angelegte Datenbank importiert werden. Dafür wird das in Kapitel 3.3.1 beschriebene Vorgehen verwendet.

Zu Beginn wird die Relation *Produkt* betrachtet. Die Produkte werden *Produkt A*, *Produkt B*, usw. genannt. In Kapitel 3.1.1 wurde bereits festgelegt, dass die Fertigung insgesamt vier Produkte produziert. Dementsprechend gibt es *Produkt A* bis *Produkt D*. Für die Artikelnummer wird ein System festgelegt, nachdem die Artikelnummer der Produkte und der Zulieferer Produkte aufgebaut ist. Dadurch soll die manuelle Erzeugung der Daten vereinfacht werden. Eine Übersicht von dem grundlegenden Aufbau jeder Identifikationsnummer kann der Abbildung 12 entnommen werden.

In der Abbildung 12 steht das *x* für eine beliebige Ziffer und das *y* für einen beliebigen Buchstaben. Die weiteren Zahlen und Buchstaben sind ein fester Bestandteil der Identifikationsnummer.

Die Artikelnummern beginnen mit einem *A*. Darauf folgt, bei den Produkten der Fertigung, die Zahlenfolge *001*. Mit einem Bindestrich getrennt folgen 3 weitere Ziffern, mit denen sich zwischen 999 Produkten unterscheiden lässt. Der Aufbau der Artikelnummern der Zuliefererprodukte unterscheidet sich, wie der Abbildung 12 entnommen werden kann, nur im ersten Teil der Identifikationsnummer. Teilenummern der gefertigten Teile beginnen mit der Ziffernfolge *001* und werden durch eine beliebige fünfstellige Ziffernfolge ergänzt. Die Teilenummern der Zuliefererteile beginnen hingegen mit *002*. Sie werden jedoch ebenso durch eine fünfstellige Ziffernfolge ergänzt. Diese Unterscheidungen werden vorgenommen, um Verwechslungen zwischen den Identifikationsnummern auszuschließen. Weiterhin muss die Lieferungsnummer erstellt werden. Auf den Anfangsbuchstaben *L* folgt eine sechsstellige Ziffernfolge. In Anbetracht des Produktionsumfangs wird bei der Hallennummer eine zweistellige Ziffernfolge verwendet. Die Roboternummern werden durch eine dreistellige Ziffernfolge aufgebaut und beginnen mit einem *R*.

Identifikationsnummer	Aufbau
ArtikelNrProdukt	A001-xxx
ArtikelNrZulieferer	A0x0-xxx
TeileNrProdukt	001-xxxxx
TeileNrZulieferer	020-xxxxx
LieferungsNr	Lxxx-xxx
HallenNr	Hxx
RoboterNr	Rxxx
PlatzNrEin	E-xx-yxx
PlatzNrAus	A-xx-yxx

Abbildung 12 Grundlegender Aufbau der Identifikationsnummern (eigene Darstellung)

Für die Platznummern des Eingangs- und Ausgangslagers wird überlegt, wie das Lager aufgebaut sein könnte und anhand des Aufbaus die Platznummer festgelegt. Ein möglicher Aufbau des Lagers ist in Abbildung 13 dargestellt. Es wird festgelegt, dass in den Lagern Regale stehen, die in Reihen angeordnet sind. Die Regale sind in jeder Reihe gleich groß. Die Aufteilung der Regale erfolgt durch Spalten, die mit Buchstaben gekennzeichnet sind, und durch Ebenen, die durch Zahlen gekennzeichnet sind. Aus diesem Lagersystem lässt sich nun eine Identifikationsnummer aufbauen, die jedes Fach identifizieren kann. Wird der Aufbau der Platznummern in Abbildung 12 betrachtet, stehen die ersten beiden Ziffern für die Reihe. Das *y* steht für die Spalte und die letzten beiden Ziffern für die Ebene. Betrachtet man das Ausgangslagers, wird dementsprechend dem Fach in Reihe 1, Spalte B und Ebene 1, die Platznummer *A-01-B01* zugeordnet. Im Folgenden wird angenommen, dass in jedem Fach nur ein Teil gelagert werden kann.

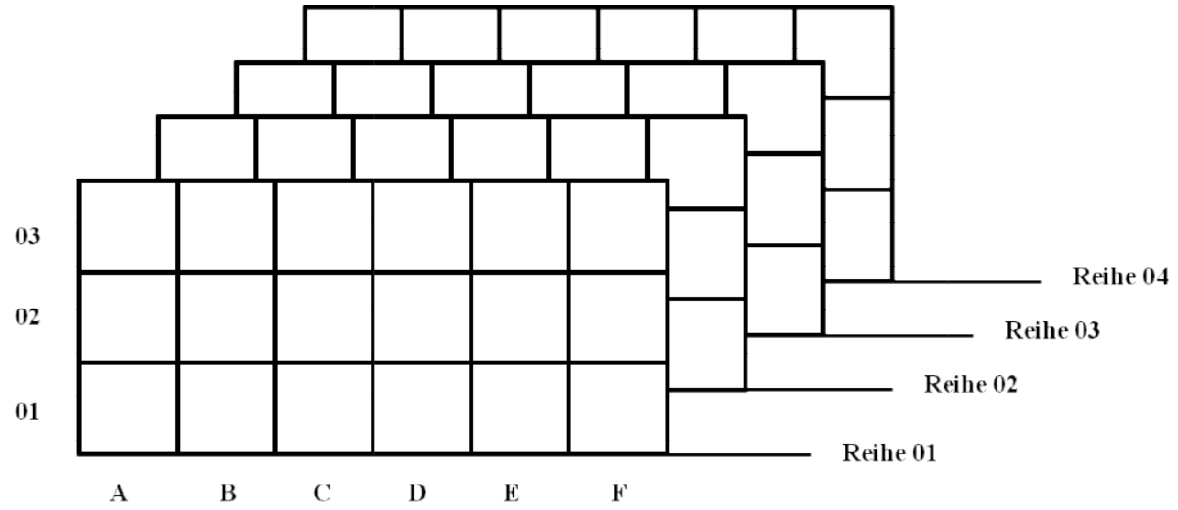

Abbildung 13 Beispielhafter Aufbau der Lager (eigene Darstellung)

Bevor mit dem grundlegenden Aufbau der Identifikationsnummern und dem Aufbau der Lager, die weiteren Daten erzeugt werden, werden weitere Annahmen getroffen, die das in Kapitel 3.1.1 definierte Umfeld ergänzen und weiterführen. Bei der Erarbeitung der weiteren Annahmen wird grundsätzlich darauf geachtet, in dem Rahmen des bereits definierten Umfeldes zu bleiben und Widersprüche zu vermeiden. Weiterhin ist der Einfluss der Annahmen auf die Datenmenge zu beachten. Das Ziel dieser Annahmen ist, die Datenmenge logisch einzugrenzen und die Datenerzeugung durch klare Strukturen zu vereinfachen.

In 3.1.1 wurde bereits festgelegt, dass die Produktpalette aus vier Produkten besteht, wobei jedes Produkt aus vier Zuliefererprodukten besteht. Daher gibt es also 16 einzelne Zuliefererprodukte. Weiterhin wird der Zeitraum festgelegt, den die Daten abbilden sollen. Es wird angenommen, dass die ersten sieben Produktionstage im Jahr 2020 betrachtet werden. Mit der Annahme, dass die Fertigung nur an Wochentagen produziert, folgt, dass der Zeitraum vom 01.01.2020 bis zum 10.01.2020 betrachtet wird. Für die Festlegung des Produktionsumfangs wird die Annahme getroffen, dass in jeder der vier Hallen zwei Fertigungsroboter stehen, die pro Tag ein Teil fertigen. Daraus folgt, dass pro Tag acht Teile von acht Robotern produziert werden. Von jedem Produkt zwei Teile. Dadurch, dass jedes produzierte Teil aus vier Zulieferteilen besteht, ist auch der Tagesbedarf an Zuliefererteilen festgelegt. Es werden 32 Zulieferteile benötigt, wobei von jedem Zuliefererprodukt genau zwei Zuliefererteile verwendet werden. Damit ist die Produktionskapazität festgelegt. Weiterhin müssen Annahmen getroffen werden, um die Lagerkapazitäten festlegen zu können. Dafür werden die Termine der Anlieferungen von Zuliefererteilen und der Lieferungen der gefertigten Teile festgelegt. Es wird angenommen, dass jeden zweiten Produktionstag neue Zuliefererteile angeliefert werden. Diese Anlieferung soll für die Produktion von zwei Tagen reichen.

Außerdem soll die Lieferung vor Produktionsbeginn ankommen, sodass die angelieferten Zuliefererteile am selben Tag für die Fertigung genutzt werden können. Auf den Betrachtungszeitraum übertragen bedeutet das, dass die erste Anlieferung am 01.01.2020 erfolgt. Bei einem Verbrauch von 32 Zuliefererteilen kommen dementsprechend pro Anlieferung 64 Teile an. Für das Eingangslager bedeutet dies eine notwendige Kapazität von 64 Plätzen. Hier wird noch ein halber Tag Puffer eingerechnet, sodass das Eingangslager eine Kapazität von 80 Plätzen aufweist. Mit Hilfe des in Abbildung 13 dargestellten Lagersystems wird festgelegt, dass Regale mit vier Ebenen und vier Spalten benutzt werden, von denen es fünf Reihen gibt.

Weiterhin müssen die Lieferungen der gefertigten Teile festgelegt werden. So wie die Anlieferung der Zuliefererteile, sollen die gefertigten Teile jeden zweiten Produktionstag abtransportiert werden. Dabei sollen alle produzierten Teile der letzten zwei Tage abtransportiert werden. Da, wie bereits festgelegt, acht Teile an einem Produktionstag gefertigt werden, muss die Lieferung 16 Teile transportieren. Weiterhin wird die Annahme getroffen, dass die Lieferung der gefertigten Teile nach Produktionsschluss beladen wird, sodass die Teile, die am selben Tag produziert wurden, mitgeliefert werden können. Übertragen auf den Betrachtungszeitraum folgt, dass die erste Lieferung von gefertigten Teilen am 02.01.2020 erfolgt und dass die Teile transportiert werden, die am 01.01.2020 und 02.01.2020 produziert wurden. Durch die Liefertermine lässt sich nun auch auf die Kapazitäten des Ausgangslagers schließen. Das Ausgangslager muss mindestens für die Produktion von zwei Tagen ausreichen. Hier wird zusätzlich noch ein Puffer von einem Tag eingerechnet. Bei acht produzierten Teilen pro Tag folgt eine Lagerkapazität von 24 Plätzen. Die 24 Plätze werden auf zwei Regale mit drei Ebenen und vier Spalten aufgeteilt.

Mit den oben getroffenen Annahmen lassen sich die Daten für die Relationen erzeugen. Die Daten werden mit Hilfe von Excel in einer CSV-Datei gespeichert. Jede Relation wird dabei separat mit Daten befüllt. Bei der Erzeugung ist darauf zu achten, dass die erzeugten Daten keinen Widerspruch zu den getätigten Annahmen darstellen.

Für die Relation der gefertigten Teile lässt sich nun die Anzahl an Datensätzen ableiten. Bei sieben Produktionstagen mit je acht gefertigten Teilen müssen 56 Teile in der Relation vorhanden sein. Hier wird aber davon ausgegangen, dass am 10.01.2020 bereits vier Teile produziert wurden. Daher werden 60 Teile der Relation Teile zugeordnet. Die weiteren Attribute der Relation werden getreu der getroffenen Annahmen nachgeführt. Durch die Annahme, dass am 10.01.2020 erst vier Teile produziert wurden, lässt sich auf die Lieferungen schließen und der Bestand des Ausgangslagers ist somit festgelegt. Auch auf die Daten der Zuliefererteile lässt sich nun schließen. Bis zum 10.01.2020 gab es vier Anlieferungen von Zuliefererteilen. Damit müssen 256 Teile in der Relation der Zuliefererteile enthalten sein. Auf Daten der weiteren Attribute wird ebenso aus den getroffenen

Annahmen geschlossen. Die Artikelnummer der Zulieferer ist beispielsweise durch den Bedarf an Produkten festgelegt und auf die Teilenummern der produzierten Teile lässt sich durch das Produktionspensum schließen. Durch den Stand der produzierten Teile und der letzten Anlieferung, ist auch der aktuelle Bestand des Eingangslagers festgelegt. Dadurch, dass die letzte Anlieferung am 09.01.2020 war und am 10.01.2020 bereits vier Teile produziert wurden, folgt, dass 16 Zuliefererteile in dem Eingangslager sein müssen. Nach gleichem Prinzip werden alle weiteren Daten erzeugt, die in Tabellenform dem Anhang 2 entnommen werden können.

Nachdem alle Daten erzeugt sind, können die Daten mit Hilfe von phpMyAdmin, wie in 3.2.1 beschrieben, importiert werden. Dabei muss jedoch darauf geachtet werden, dass die Zellen, die als Nullwerte in die Datenbank eingefügt werden sollen, mit NULL befüllt werden müssen. Weiterhin wird beim Importieren, wie in 3.3.1 bereits erläutert, die in 3.2.2 verwendete Reihenfolge benutzt. Schließlich ist darauf zu achten, dass ein Datum mit dem Jahr, dann dem Monat und dann dem Tag angezeigt wird. Ansonsten kann es zu einem fehlerhaften Import kommen. Die damit befüllte Datenbank kann durch die SQL-Datei, welche dieser Projektarbeit beigefügt ist, mit Hilfe von phpMyAdmin importiert werden. Dafür kann in phpMyAdmin eine neue Datenbank mit dem Datenbanknamen *produktionsdb* und der Kollation *latin1_swedish_ci* angelegt werden. In diese kann anschließend die SQL-Datei importiert werden.

4. Konzipierung von Abfrageszenarien zu Lehrzwecken

Das Ziel der Konzipierung von Abfrageszenarien in diesem Kapitel ist, den Teilnehmenden einer Lehrveranstaltung die Möglichkeit zu geben, die in Kapitel 3 erstellte Datenbank zu Übungszwecken zu nutzen. Damit die Studierenden die Abfrageszenarien effektiv nutzen können, muss bei der Konzipierung der Wissensstand der Studierenden beachtet werden. Daher wird vor der Konzipierung der Abfrageszenarien in Kapitel 4.2, der Wissensstand der Studierenden in Kapitel 4.1 erfasst. Die Konzipierung der Abfrageszenarien umfasst neben dem eigentlichen Erstellen der Szenarien auch die Darstellung der Lösungswege. Weiterhin werden die Abfrageszenarien in Kapitel 4.2.2 klassifiziert.

4.1. Wissensstand der Teilnehmer der Lehrveranstaltung

Die in Kapitel 3 erstellte MySQL-Datenbank soll von den Teilnehmenden der Lehrveranstaltung „Informationsaustausch produzierender Unternehmen“ zu Übungszwecken genutzt werden können. Die dafür zu erstellenden Abfrageszenarien sollen, durch die in der Lehrveranstaltung vermittelten Inhalte zu lösen sein. Die Lehrveranstaltung wird in dem Master-Studiengang Maschinenbau als Wahlpflichtmodul angeboten und kann von den Studierenden der Profile „Produktionstechnik“, „Technische Betriebsführung“ und „IT in Produktion und Logistik“ belegt werden. Das Modul wird von dem Fachgebiet ITPL gehalten und befasst sich mit dem Informationsaustausch "im Wertschöpfungsnetz entlang der Kette der Produktentstehung sowie im Zulieferer- und Distributionsnetz" (Teichmann, S. 55). Ein Teil des Moduls ist das Thema der Datenbanken. In diesem Teil werden die Studierenden in die Begriffswelt der Datenbanken eingeführt und besonders die relationalen Datenbanken behandelt. An dieser Stelle wird die Datenbanksprache SQL vorgestellt. Um Abfrageszenarien konzipieren zu können, welche zu dem Wissenstand der Studierendenden passen, wird sich im Folgenden auf die in der Lehrveranstaltung vermittelten SQL-Befehle beschränkt. Auch wird auf weitere Lehrinhalte des Moduls hier nicht weiter eingegangen.

Bereits in Kapitel 2.1.3 wurde die Datenbanksprache SQL eingeführt. Dafür wurde eine Auswahl von SQL Befehlen näher erläutert, welche sich an die in der Lehrveranstaltung verwendeten Befehle orientierte. Daher kann diese Auswahl für die Konzipierung der Abfrageszenarien hinzugezogen werden. Dafür wird im Folgenden die Übersicht in Abbildung 8 verwendet.

4.2. Konzipierung der Abfrageszenarien

Für die Konzipierung der Abfrageszenarien, welche zu Übungszwecken genutzt werden sollen, werden zu Beginn, anhand des in Kapitel 3.1.1 definierten Umfelds, Szenarien erstellt und mit einem möglichen Lösungsweg dokumentiert. Anschließend werden die Abfrageszenarien in Kapitel 4.2.2 eingeordnet.

4.2.1. Erstellen der Abfrageszenarien und der Lösungswege

Bei der Konzipierung werden Abfragen gesucht, für dessen Lösung die Befehle aus der Abbildung 8 verwendet werden können. Dafür werden die Befehle oder Befehlskombinationen einzeln betrachtet und nach einem passenden Abfrageszenario gesucht. Dann wird passend zu dem Szenario ein möglicher Lösungsweg in Form von einem SQL-Code dargestellt. Dabei sollen die Abfrageszenarien schlüssig zu dem Umfeld passen. Dies führt dazu, dass dem Anwender der Szenarien die grundlegenden Strukturen des Umfelds bekannt sein müssen. Außerdem ist anzumerken, dass die hier ausgewählten Szenarien lediglich eine Möglichkeit darstellen, die Anwendung der, in der Lehrveranstaltung vorkommenden, SQL-Befehle anhand der erstellten Datenbank zu üben.

Für das erste Szenario soll der Befehl SELECT verwendet werden.

- Abfrageszenario 1:

  Geben Sie alle Zuliefererteile mit der zugehörigen Platznummer aus, die sich zum jetzigen Zeitpunkt im Eingangslager befinden.

  Lösungsweg 1:

```
SELECT eingangslager.PlatzNrEin, eingangslager.TeileNrZulieferer
FROM eingangslager
WHERE eingangslager.TeileNrZulieferer IS NOT NULL
```

Szenario 2 soll für die Übung von SELECT und DISTINCT verwendet werden.

- Abfrageszenario 2:

  Geben Sie die Tage aus, an denen Zuliefererteile angeliefert wurden, ohne ein Datum mehrfach anzuzeigen.

  Lösungsweg 2:

```
SELECT DISTINCT zuliefererteile.Anlieferung
FROM zuliefererteile
```

Für den Lösungsweg von Szenario 3 soll SELECT und COUNT verwendet werden.

- Abfrageszenario 3:

  Geben Sie die Anzahl der Teile aus, die in der Lieferung mit der Lieferungsnummer L780-002 transportiert wurden und dem Produkt mit der Artikelnummer A001-029 zugeordnet sind.

  Lösungsweg 3:

```
SELECT teile.ArtikelNrProdukt, teile.LieferungsNr,
COUNT(teile.TeileNrProdukt)Anzahl
FROM teile
WHERE teile.LieferungsNr = "L780-002" AND teile.ArtikelNrProdukt = "A001-029"
```

Durch das vierte Szenario soll der Befehl ORDER BY angewendet werden können.

- Abfrageszenario 4:

  Geben Sie alle Teile aus, die von dem Fertigungsroboter R085 gefertigt wurden und ordnen Sie die Teile nach dem Produktionsdatum. Das zuletzt produzierte Teil soll dabei als erstes angezeigt werden.

  Lösungsweg 4:

```
SELECT teile.RoboterNr, teile.Produktionsdatum, teile.TeileNrProdukt
FROM teile
WHERE teile.RoboterNr = "R085"
ORDER BY (teile.Produktionsdatum) DESC
```

Die Anwendung des Befehls JOIN soll durch das fünfte Szenario geübt werden können. Hier wird ein Szenario für den INNER JOIN gewählt.

- Abfrageszenario 5:

  Geben Sie die Platznummern des Ausgangslagers aus, die Teile von dem Produkt A001-176 lagern. Verknüpfen Sie dafür die Relation Ausgangslager und Teile mit dem INNER JOIN.

  Lösungsweg 5:

```
SELECT ausgangslager.PlatzNrAus, teile.ArtikelNrProdukt
FROM ausgangslager
INNER JOIN teile
ON ausgangslager.TeileNrProdukt = teile.TeileNrProdukt
WHERE teile.ArtikelNrProdukt = "A001-176"
```

Diese ersten fünf Abfragen können dazu dienen, die Anwendung verschiedener SELECT-Abfragen zu üben. Die nächsten drei Szenarien sollen hingehen eine Möglichkeit für die Übung der Befehle INSERT, UPDATE und DELETE darstellen.

- Abfrageszenario 6:

  Es wird ein neuer Fertigungsroboter mit der Roboternummer R001 angeschafft, der in der Halle H02 Teile produzieren soll. Fügen Sie den Roboter in die Relation *Fertigungsroboter* ein.

  Lösungsweg 6:

```
INSERT INTO fertigungsroboter (RoboterNr, HallenNr)
VALUES ("R001","H02")
```

- Abfrageszenario 7:

  Der Fertigungsroboter R005 wird aussortiert und ist keiner Halle mehr zugeordnet. Aktualisieren Sie den Datensatz des Roboters.

  Lösungsweg 7:

```
UPDATE fertigungsroboter
SET HallenNr = Null
WHERE RoboterNr = "R005"
```

- Abfrageszenario 8:

  Im Ausgangslager soll der Platz A02-D03 dauerhaft anderweitig genutzt werden. Löschen Sie daher die Platznummer aus der Datenbank.

  Lösungsweg 8:

```
DELETE FROM ausgangslager
WHERE PlatzNrAus = "A-02-D03"
```

Durch die eben angeführten Abfrageszenarien lassen sich ein Teil der SQL-Befehle, anhand der vorliegende Datenbank, anwenden. Bei den Abfrageszenarien wurde darauf geachtet, keine Abfragen zu konzipieren, welche in mehreren Schritten ausgeführt werden müssen. Dadurch soll es ermöglicht werden, jede Abfrage separat und unabhängig von weiteren Abfragen zu üben. Um ausführlichere Abfragen zu üben, kann es jedoch sinnvoll sein, neben eben aufgeführten Szenarien, weitreichendere Szenarien zu konzipieren, für dessen Lösung mehrere Schritte notwendig sind. Weiterhin können die Ergebnisse, welche durch die Lösungswege erreicht werden, dem Anhang 3 entnommen werden. Die dauerhaften Änderungen durch Szenario 6, Szenario 7 und Szenario 8 können den separat beigefügten Datenbanken entnommen werden.

4.2.2. Klassifizierung und Einordnung der Abfrageszenarien

Das Ziel einer Klassifizierung der in Kapitel 4.2.1 konzipierten Abfragen ist, den Studierenden, welche die Abfrageszenarien eigenständig bearbeiten sollen, einen Anhaltspunkt zu geben in welcher Reihenfolge die Abfragen bearbeitet werden können. Dabei bietet es sich an, Abfragen, welche eine geringe Komplexität aufweisen, zu Beginn zu bearbeiten. Im Rahmen dieser Projektarbeit werden die Abfragen anhand der, für die Abfrage notwendigen, Befehle und Bedingungen eingeordnet. Dafür wird die Anzahl an einzelnen Befehlen und Bedingungen, die bei einer Abfrage verwendet werden, festgestellt und miteinander verglichen. Besteht ein Befehl aus mehreren Schlüsselwörtern, wird jedes Schlüsselwort gezählt. Dabei ist jedoch deutlich anzumerken, dass die Klassifizierung lediglich als Anhaltspunkt genutzt werden kann und die Abfragen nicht eindeutig nach einem Schwierigkeitsgrad geordnet werden. Das folgt daraus, dass das Empfinden, ob eine Abfrage schwierig zu lösen ist, subjektiv und nicht einheitlich bewertet werden kann.

Kennzahl	Szenario	Abfrage
2	6	`INSERT INTO fertigungsroboter (RoboterNr, HallenNr)` `VALUES ("R001","H02")`
2	8	`DELETE FROM ausgangslager` `WHERE PlatzNrAus = "A02-D03"`

3	1	`SELECT eingangslager.PlatzNrEin,`
		`eingangslager.TeileNrZulieferer`
		`FROM eingangslager`
		`WHERE eingangslager.TeileNrZulieferer IS NOT NULL`

3	2	`SELECT DISTINCT zuliefererteile.Anlieferung` `FROM zuliefererteile`

3	7	`UPDATE fertigungsroboter`
		`SET HallenNr = Null`
		`WHERE RoboterNr = "R005"`

4	4	`SELECT teile.RoboterNr, teile.Produktionsdatum,`
		`teile.TeileNrProdukt`
		`FROM teile`
		`WHERE teile.RoboterNr = "R085"`
		`ORDER BY (teile.Produktionsdatum) DESC`

5	3	`SELECT teile.ArtikelNrProdukt, teile.LieferungsNr,`
		`COUNT(teile.TeileNrProdukt)Anzahl`
		`FROM teile`
		`WHERE teile.LieferungsNr = "L780-002" AND`
		`teile.ArtikelNrProdukt = "A001-029"`

5	5	`SELECT ausgangslager.PlatzNrAus, teile.ArtikelNrProdukt`
		`FROM ausgangslager`
		`INNER JOIN teile`
		`ON ausgangslager.TeileNrProdukt = teile.TeileNrProdukt`
		`WHERE teile.ArtikelNrProdukt = "A001-176"`

Abbildung 14 Einordnung der Abfrageszenarien (eigene Darstellung)

In Abbildung 14 ist die Einordnung der Abfrageszenarien dargestellt. Der Begriff *Kennzahl* wird hier für die Anzahl der Schlüsselworte verwendet. Bei dem Zählen werden Schlüsselwörter, wie INSERT INTO, als ein Schlüsselwort betrachten. Anders wird beispielsweise SELECT DISTINCT behandelt. DISTINCT ist hier eine Erweiterung des SELECT Befehls, welcher, wie Szenario 1 entnommen werden kann, auch ohne den Zusatz DISTNCT verwendet werden kann. Daher wird die Kennzahl, durch den Befehl SELECT DISTINCT, um zwei Schlüsselwörter erweitert. Weiterhin wird die Kennzahl für jede Bedingung um Eins erhöht. Wird dieses Vorgehen auf die Abfragen angewendet, lassen sich die in Abbildung 14 zusammengefassten Kennzahlen festlegen. Die Zuordnung der Kennzahlen zu den einzelnen Abfrageszenarien erfolgt durch die in 4.2.1 verwendete Nummerierung der Szenarien. Weiterhin wird die Abfrage, auf die sich bezogen wird, als SQL-Code dargestellt. Die Abfragen sind nach aufsteigender Kennzahl eingeordnet. Damit kann die Abbildung 14 als Anhaltspunkt verwendet werden, um die Reihenfolge, in der die Abfrageszenarien bearbeitet werden, festzulegen.

5. Zusammenfassung und Ausblick

Das Ziel der vorliegenden Projektarbeit war die Erstellung einer SQL-Datenbank, welche ein produktionstechnisches Umfeld abbildet. Mit Beispieldaten befüllt, soll die Datenbank Studierenden eine Möglichkeit bieten, Inhalte aus einer Lehrveranstaltung vertiefen zu können. Somit soll die in der Lehrveranstaltung eingeführte Datenbanksprache SQL, anhand der erstellten Datenbank, angewendet werden können.

Die Entwicklung der Datenbank stützte sich vorwiegend auf die Rechercheergebnisse im Bereich der relationalen Datenbanken, die durch die Einführung der Produktionstechnik und des Anwendungsprogramms phpMyAdmin, ergänzt wurden. Vor der eigentlichen Entwicklung der Datenbank musste ein fiktives produktionstechnisches Umfeld definiert werden, das als Grundlage für die Erstellung der Datenbank verwendet wurde. Es folgte die Überführung des Umfeldes in ein ERM. Für die erfolgreiche Überführung waren klar definierte Strukturen innerhalb des Umfeldes von Bedeutung, sodass Entitäten und Beziehungen eindeutig zusammengetragen werden konnten. Das ERM konnte anschließend in Relationen überführt werden und mit Hilfe des eingerichteten Anwendungsprogramm phpMyAdmin implementiert werden. Um die damit implementierte Datenbank mit Beispieldaten zu befüllen, wurden in dem bestehenden produktionstechnischen Umfeld weitere Annahmen getroffen. So wurden für die Erzeugung von Daten, die innerhalb des Umfeldes schlüssig sein sollten, der Aufbau von Identifikationsnummern und ein Lagersystem festgelegt. Mit Hilfe von CSV-Dateien und phpMyAdmin konnten die Daten schließlich in die Datenbank importiert werden. Um Möglichkeiten für die Nutzung der somit erstellten Datenbank aufzuzeigen, wurden beispielhafte Abfrageszenarien konzipiert. Dabei wurde der, durch die Lehrveranstaltung vermittelte, Wissensstand hinzugezogen.

Abschließend kann angeführt werden, dass die erstellte Datenbank eines fiktiven produktionstechnischen Umfelds, eine Möglichkeit darstellen kann, die Anwendung der Datenbanksprache SQL zu üben. Die in Kapitel 4 konzipierten Abfrageszenarien deuten dabei lediglich auf einen möglichen Nutzen der Datenbank für Übungszwecke hin und decken nicht alle möglichen Abfrageszenarien innerhalb des Umfelds auf. Um die Anwendung von umfangreichenden Abfragen zu vertiefen, kann es sinnvoll sein über die bereits erstellen Szenarien hinaus, weitere mögliche Szenarien dem Umfeld zu entnehmen.

Abkürzungsverzeichnis

DBMS	Datenbank Management System
SQL	Structured Query Language
ITPL	IT in Produktion und Logistik
ERM	Entity-Relationship-Modell
DDL	Data Definition Language
DML	Data Manipulation Language
DCL	Data Control Language

Abbildungsverzeichnis

Literaturverzeichnis

Geisler, Frank (2014): Datenbanken. Grundlagen und Design. 5., aktualisierte und erw. Aufl. Heidelberg: mitp. Online verfügbar unter https://link.springer.com/content/pdf/10.1007%2F978-3-642-14502-5.pdf.

Matthiessen, Günter; Unterstein, Michael (2007): Relationale Datenbanken und Standard-SQL. Konzepte der Entwicklung und Anwendung. 1. Aufl. München, Boston, Mass. u.a.: Addison-Wesley.

Michael Rüttger (2019): Von MySQL AB bis Oracle: MySQL Grundkurs. Online verfügbar unter https://www.linkedin.com/learning/mysql-grundkurs-2/von-mysql-ab-bis-oracle?u=81842386, zuletzt aktualisiert am 12.03.2020, zuletzt geprüft am 12.03.2020.

phpMyAdmin (2020): phpMyAdmin. Online verfügbar unter https://www.phpmyadmin.net/, zuletzt aktualisiert am 12.03.2020, zuletzt geprüft am 12.03.2020.

Rabe, M. (2019): Informationsaustausch produzierender Unternehmen. WS 2019/20, Vorlesung 12 - SQL. Hg. v. Fachgebiet IT in Produktion und Logistik.

Saake, Gunter; Sattler, Kai-Uwe; Heuer, Andreas (2018): Datenbanken Konzepte und Sprachen. Sechste Auflage. Frechen: mitp.

Sauer, Hermann; Grieger, Klaus (1998): Relationale Datenbanken. Theorie und Praxis. 4., aktualisierte u. erw. Aufl. München: Addison-Wesley (Datenbanken).

Schicker, Edwin (2017): Datenbanken und SQL. Eine praxisorientierte Einführung mit Anwendungen in Oracle, SQL Server und MySQL. 5., aktualisierte und erweiterte Auflage. Wiesbaden: Springer Vieweg (Informatik & Praxis). Online verfügbar unter https://www.degruyter.com/downloadpdf/books/9783110413908/9783110413908-003/9783110413908-003.pdf, zuletzt geprüft am 11.10.2019.

Schuh, Günther; Kampker, Achim (2011): Strategie und Management produzierender Unternehmen. Handbuch Produktion und Management 1. 2., vollst. neu bearb. und erw. Aufl. Berlin, Heidelberg: Springer-Verlag Berlin Heidelberg (VDI-Buch). Online verfügbar unter http://dx.doi.org/10.1007/978-3-642-14502-5, zuletzt geprüft am 14.03.2020.

Skolaut, Werner (2014): Maschinenbau. Berlin, Heidelberg: Springer Berlin Heidelberg.

Steiner, René (2017): Grundkurs Relationale Datenbanken. Einführung in die Praxis der Datenbankentwicklung für Ausbildung, Studium und IT-Beruf. 9. Aufl. 2017. Wiesbaden: Springer Fachmedien Wiesbaden; Imprint Springer Vieweg.

Teichmann: Modulhandbuch_Master_Maschinenbau_WS19_07_19. Online verfügbar unter https://www.mb.tu-dortmund.de/cms/Medienpool/NEU_Studium/Modulhandbuch_Master_Maschinenbau_WS19_07_19.pdf, zuletzt geprüft am 28.10.2019.

Vossen, Gottfried; Lausen, Georg (2018): Objekt-orientierte Datenbanken: Modelle und Sprachen: De Gruyter.

XAMPP (2020). Online verfügbar unter https://www.apachefriends.org/de/index.html, zuletzt aktualisiert am 02.04.2020, zuletzt geprüft am 03.04.2020.

Anhang 1: Quellcode für die Implementierung der Relationen

```
CREATE TABLE Produkt(
    ArtikelNrProdukt CHAR(8) PRIMARY KEY,
    Name Varchar(200)
    );

CREATE TABLE ZuliefererProdukte(
    ArtikelNrZulieferer CHAR(8) PRIMARY KEY,
    Name VARCHAR(200),
    ArtikelNrProdukt CHAR(8) NOT NULL,
    FOREIGN KEY(ArtikelNrProdukt) REFERENCES Produkt(ArtikelNrProdukt)
    );

CREATE TABLE Lieferung(
    LieferungsNr char(8) PRIMARY KEY,
    Lieferdatum DATE
    )

CREATE TABLE Fertigungshalle(
    HallenNr char(3) PRIMARY KEY,
    ArtikelNrProdukt char(8)NOT NULL,
    FOREIGN KEY(ArtikelNrProdukt) REFERENCES Produkt(ArtikelNrProdukt)
    );

CREATE TABLE Fertigungsroboter(
    RoboterNr char(4) PRIMARY KEY,
    HallenNr char(3),
    FOREIGN KEY (HallenNr) REFERENCES fertigungshalle(HallenNr)
    );

CREATE TABLE Teile(
    TeileNrProdukt char(9) PRIMARY KEY,
    Produktionsdatum DATE,
    ArtikelNrProdukt char(8) NOT NULL,
    RoboterNr char(4) NOT NULL,
    LieferungsNr char(8),
    FOREIGN KEY(ArtikelNrProdukt) REFERENCES Produkt(ArtikelNrProdukt),
    FOREIGN KEY(RoboterNr) REFERENCES Fertigungsroboter(RoboterNr),
    FOREIGN KEY(LieferungsNr) REFERENCES Lieferung(LieferungsNr)
    );

CREATE TABLE Ausgangslager(
    PlatzNrAus char(8) PRIMARY KEY,
    TeileNrProdukt char(9) UNIQUE,
    FOREIGN KEY(TeileNrProdukt) REFERENCES Teile(TeileNrProdukt)
    );

CREATE TABLE ZulieferTeile(
    TeileNrZulieferer char(9) NOT NULL PRIMARY KEY,
    Anlieferung DATE,
    ArtikelNrZulieferer char(8) NOT NULL,
    TeileNrProdukt char(9),
    FOREIGN KEY(ArtikelNrZulieferer) REFERENCES ZuliefererProdukte(ArtikelNrZulieferer),
    FOREIGN KEY(TeileNrProdukt) REFERENCES Teile(TeileNrProdukt)
    );

CREATE TABLE Eingangslager(
    PlatzNrEin CHAR(8) PRIMARY KEY,
    TeileNrZulieferer char(9) UNIQUE,
    FOREIGN KEY(TeileNrZulieferer) REFERENCES ZulieferTeile(TeileNrZulieferer)
    );
```

Anhang 2: Die Datensätze für die Datenbank

Produkt	
ArtikelNrProdukt	Name
A001-234	Produkt A
A001-176	Produkt B
A001-329	Produkt C
A001-029	Produkt D

ZuliefererProdukte		
ArtikelNrZulieferer	Name	ArtikelNrProdukt
A020-010	Komponente A1	A001-234
A030-010	Komponente A2	A001-234
A020-020	Komponente A3	A001-234
A060-040	Komponente A4	A001-234
A040-210	Komponente B1	A001-176
A050-010	Komponente B2	A001-176
A060-120	Komponente B3	A001-176
A030-080	Komponente B4	A001-176
A030-140	Komponente C1	A001-329
A040-060	Komponente C2	A001-329
A050-070	Komponente C3	A001-329
A020-090	Komponente C4	A001-329
A060-070	Komponente D1	A001-029
A050-030	Komponente D2	A001-029
A020-030	Komponente D3	A001-029
A030-130	Komponente D4	A001-029

Lieferung	
LieferungsNr	Lieferdatum
L780-001	2020.01.02
L780-002	2020.01.06
L780-003	2020.01.08

Fertigungshalle	
HallenNr	ArtikelNrProdukt
H01	A001-029
H02	A001-329
H03	A001-234
H04	A001-176

Anhang 2: Die Datensätze für die Datenbank

Fertigungsroboter	
RoboterNr	HallenNr
R031	H04
R002	H01
R086	H02
R085	H03
R023	H04
R030	H01
R005	H02
R006	H03

Teile				
TeileNrProdukt	Produktionsdatum	ArtikelNrProdukt	RoboterNr	LieferungsNr
001-00001	2020.01.01	A001-176	R031	L780-001
001-00002	2020.01.01	A001-029	R002	L780-001
001-00003	2020.01.01	A001-329	R086	L780-001
001-00004	2020.01.01	A001-234	R085	L780-001
001-00005	2020.01.01	A001-176	R023	L780-001
001-00006	2020.01.01	A001-029	R030	L780-001
001-00007	2020.01.01	A001-329	R005	L780-001
001-00008	2020.01.01	A001-234	R006	L780-001
001-00009	2020.01.02	A001-176	R031	L780-001
001-00010	2020.01.02	A001-029	R002	L780-001
001-00011	2020.01.02	A001-329	R086	L780-001
001-00012	2020.01.02	A001-234	R085	L780-001
001-00013	2020.01.02	A001-176	R023	L780-001
001-00014	2020.01.02	A001-029	R030	L780-001
001-00015	2020.01.02	A001-329	R005	L780-001
001-00016	2020.01.02	A001-234	R006	L780-001
001-00017	2020.01.03	A001-176	R031	L780-002
001-00018	2020.01.03	A001-029	R002	L780-002
001-00019	2020.01.03	A001-329	R086	L780-002
001-00020	2020.01.03	A001-234	R085	L780-002
001-00021	2020.01.03	A001-176	R023	L780-002
001-00022	2020.01.03	A001-029	R030	L780-002
001-00023	2020.01.03	A001-329	R005	L780-002
001-00024	2020.01.03	A001-234	R006	L780-002
001-00025	2020.01.06	A001-176	R031	L780-002
001-00026	2020.01.06	A001-029	R002	L780-002
001-00027	2020.01.06	A001-329	R086	L780-002
001-00028	2020.01.06	A001-234	R085	L780-002
001-00029	2020.01.06	A001-176	R023	L780-002
001-00030	2020.01.06	A001-029	R030	L780-002
001-00031	2020.01.06	A001-329	R005	L780-002

Anhang 2: Die Datensätze für die Datenbank

001-00032	2020.01.06	A001-234	R006	L780-002
001-00033	2020.01.07	A001-176	R031	L780-003
001-00034	2020.01.07	A001-029	R002	L780-003
001-00035	2020.01.07	A001-329	R086	L780-003
001-00036	2020.01.07	A001-234	R085	L780-003
001-00037	2020.01.07	A001-176	R023	L780-003
001-00038	2020.01.07	A001-029	R030	L780-003
001-00039	2020.01.07	A001-329	R005	L780-003
001-00040	2020.01.07	A001-234	R006	L780-003
001-00041	2020.01.08	A001-176	R031	L780-003
001-00042	2020.01.08	A001-029	R002	L780-003
001-00043	2020.01.08	A001-329	R086	L780-003
001-00044	2020.01.08	A001-234	R085	L780-003
001-00045	2020.01.08	A001-176	R023	L780-003
001-00046	2020.01.08	A001-029	R030	L780-003
001-00047	2020.01.08	A001-329	R005	L780-003
001-00048	2020.01.08	A001-234	R006	L780-003
001-00049	2020.01.09	A001-176	R031	NULL
001-00050	2020.01.09	A001-029	R002	NULL
001-00051	2020.01.09	A001-329	R086	NULL
001-00052	2020.01.09	A001-234	R085	NULL
001-00053	2020.01.09	A001-176	R023	NULL
001-00054	2020.01.09	A001-029	R030	NULL
001-00055	2020.01.09	A001-329	R005	NULL
001-00056	2020.01.09	A001-234	R006	NULL
001-00057	2020.01.10	A001-176	R031	NULL
001-00058	2020.01.10	A001-029	R002	NULL
001-00059	2020.01.10	A001-329	R086	NULL
001-00060	2020.01.10	A001-234	R085	NULL

Ausgangslager	
PlatzNrAus	TeileNrProdukt
A-01-A01	001-00049
A-01-A02	001-00050
A-01-A03	001-00051
A-01-B01	001-00052
A-01-B02	001-00053
A-01-B03	001-00054
A-01-C01	001-00055
A-01-C02	001-00056
A-01-C03	001-00057
A-01-D01	001-00058
A-01-D02	001-00059
A-01-D03	001-00060
A-02-A01	NULL

Anhang 2: Die Datensätze für die Datenbank

A-02-A02	NULL
A-02-A03	NULL
A-02-B01	NULL
A-02-B02	NULL
A-02-B03	NULL
A-02-C01	NULL
A-02-C02	NULL
A-02-C03	NULL
A-02-D01	NULL
A-02-D02	NULL
A-02-D03	NULL

ZuliefererTeile			
TeileNrZulieferer	Anlieferung	ArtikelNrZulieferer	TeileNrProdukt
002-00001	2020.01.01	A020-010	001-00004
002-00002	2020.01.01	A030-010	001-00004
002-00003	2020.01.01	A020-020	001-00004
002-00004	2020.01.01	A060-040	001-00004
002-00005	2020.01.01	A040-210	001-00001
002-00006	2020.01.01	A050-010	001-00001
002-00007	2020.01.01	A060-120	001-00001
002-00008	2020.01.01	A030-080	001-00001
002-00009	2020.01.01	A030-140	001-00003
002-00010	2020.01.01	A040-060	001-00003
002-00011	2020.01.01	A050-070	001-00003
002-00012	2020.01.01	A020-090	001-00003
002-00013	2020.01.01	A060-070	001-00002
002-00014	2020.01.01	A050-030	001-00002
002-00015	2020.01.01	A020-030	001-00002
002-00016	2020.01.01	A030-130	001-00002
002-00017	2020.01.01	A020-010	001-00008
002-00018	2020.01.01	A030-010	001-00008
002-00019	2020.01.01	A020-020	001-00008
002-00020	2020.01.01	A060-040	001-00008
002-00021	2020.01.01	A040-210	001-00005
002-00022	2020.01.01	A050-010	001-00005
002-00023	2020.01.01	A060-120	001-00005
002-00024	2020.01.01	A030-080	001-00005
002-00025	2020.01.01	A030-140	001-00007
002-00026	2020.01.01	A040-060	001-00007
002-00027	2020.01.01	A050-070	001-00007
002-00028	2020.01.01	A020-090	001-00007
002-00029	2020.01.01	A060-070	001-00006
002-00030	2020.01.01	A050-030	001-00006
002-00031	2020.01.01	A020-030	001-00006

Anhang 2: Die Datensätze für die Datenbank

002-00032	2020.01.01	A030-130	001-00006
002-00033	2020.01.01	A020-010	001-00012
002-00034	2020.01.01	A030-010	001-00012
002-00035	2020.01.01	A020-020	001-00012
002-00036	2020.01.01	A060-040	001-00012
002-00037	2020.01.01	A040-210	001-00009
002-00038	2020.01.01	A050-010	001-00009
002-00039	2020.01.01	A060-120	001-00009
002-00040	2020.01.01	A030-080	001-00009
002-00041	2020.01.01	A030-140	001-00011
002-00042	2020.01.01	A040-060	001-00011
002-00043	2020.01.01	A050-070	001-00011
002-00044	2020.01.01	A020-090	001-00011
002-00045	2020.01.01	A060-070	001-00010
002-00046	2020.01.01	A050-030	001-00010
002-00047	2020.01.01	A020-030	001-00010
002-00048	2020.01.01	A030-130	001-00010
002-00049	2020.01.01	A020-010	001-00016
002-00050	2020.01.01	A030-010	001-00016
002-00051	2020.01.01	A020-020	001-00016
002-00052	2020.01.01	A060-040	001-00016
002-00053	2020.01.01	A040-210	001-00013
002-00054	2020.01.01	A050-010	001-00013
002-00055	2020.01.01	A060-120	001-00013
002-00056	2020.01.01	A030-080	001-00013
002-00057	2020.01.01	A030-140	001-00015
002-00058	2020.01.01	A040-060	001-00015
002-00059	2020.01.01	A050-070	001-00015
002-00060	2020.01.01	A020-090	001-00015
002-00061	2020.01.01	A060-070	001-00014
002-00062	2020.01.01	A050-030	001-00014
002-00063	2020.01.01	A020-030	001-00014
002-00064	2020.01.01	A030-130	001-00014
002-00065	2020.01.03	A020-010	001-00020
002-00066	2020.01.03	A030-010	001-00020
002-00067	2020.01.03	A020-020	001-00020
002-00068	2020.01.03	A060-040	001-00020
002-00069	2020.01.03	A040-210	001-00017
002-00070	2020.01.03	A050-010	001-00017
002-00071	2020.01.03	A060-120	001-00017
002-00072	2020.01.03	A030-080	001-00017
002-00073	2020.01.03	A030-140	001-00019
002-00074	2020.01.03	A040-060	001-00019
002-00075	2020.01.03	A050-070	001-00019
002-00076	2020.01.03	A020-090	001-00019
002-00077	2020.01.03	A060-070	001-00018

Anhang 2: Die Datensätze für die Datenbank

002-00078	2020.01.03	A050-030	001-00018
002-00079	2020.01.03	A020-030	001-00018
002-00080	2020.01.03	A030-130	001-00018
002-00081	2020.01.03	A020-010	001-00024
002-00082	2020.01.03	A030-010	001-00024
002-00083	2020.01.03	A020-020	001-00024
002-00084	2020.01.03	A060-040	001-00024
002-00085	2020.01.03	A040-210	001-00021
002-00086	2020.01.03	A050-010	001-00021
002-00087	2020.01.03	A060-120	001-00021
002-00088	2020.01.03	A030-080	001-00021
002-00089	2020.01.03	A030-140	001-00023
002-00090	2020.01.03	A040-060	001-00023
002-00091	2020.01.03	A050-070	001-00023
002-00092	2020.01.03	A020-090	001-00023
002-00093	2020.01.03	A060-070	001-00022
002-00094	2020.01.03	A050-030	001-00022
002-00095	2020.01.03	A020-030	001-00022
002-00096	2020.01.03	A030-130	001-00022
002-00097	2020.01.03	A020-010	001-00028
002-00098	2020.01.03	A030-010	001-00028
002-00099	2020.01.03	A020-020	001-00028
002-00100	2020.01.03	A060-040	001-00028
002-00101	2020.01.03	A040-210	001-00025
002-00102	2020.01.03	A050-010	001-00025
002-00103	2020.01.03	A060-120	001-00025
002-00104	2020.01.03	A030-080	001-00025
002-00105	2020.01.03	A030-140	001-00027
002-00106	2020.01.03	A040-060	001-00027
002-00107	2020.01.03	A050-070	001-00027
002-00108	2020.01.03	A020-090	001-00027
002-00109	2020.01.03	A060-070	001-00026
002-00110	2020.01.03	A050-030	001-00026
002-00111	2020.01.03	A020-030	001-00026
002-00112	2020.01.03	A030-130	001-00026
002-00113	2020.01.03	A020-010	001-00032
002-00114	2020.01.03	A030-010	001-00032
002-00115	2020.01.03	A020-020	001-00032
002-00116	2020.01.03	A060-040	001-00032
002-00117	2020.01.03	A040-210	001-00029
002-00118	2020.01.03	A050-010	001-00029
002-00119	2020.01.03	A060-120	001-00029
002-00120	2020.01.03	A030-080	001-00029
002-00121	2020.01.03	A030-140	001-00031
002-00122	2020.01.03	A040-060	001-00031
002-00123	2020.01.03	A050-070	001-00031

Anhang 2: Die Datensätze für die Datenbank

002-00124	2020.01.03	A020-090	001-00031
002-00125	2020.01.03	A060-070	001-00030
002-00126	2020.01.03	A050-030	001-00030
002-00127	2020.01.03	A020-030	001-00030
002-00128	2020.01.03	A030-130	001-00030
002-00129	2020.01.07	A020-010	001-00036
002-00130	2020.01.07	A030-010	001-00036
002-00131	2020.01.07	A020-020	001-00036
002-00132	2020.01.07	A060-040	001-00036
002-00133	2020.01.07	A040-210	001-00033
002-00134	2020.01.07	A050-010	001-00033
002-00135	2020.01.07	A060-120	001-00033
002-00136	2020.01.07	A030-080	001-00033
002-00137	2020.01.07	A030-140	001-00035
002-00138	2020.01.07	A040-060	001-00035
002-00139	2020.01.07	A050-070	001-00035
002-00140	2020.01.07	A020-090	001-00035
002-00141	2020.01.07	A060-070	001-00034
002-00142	2020.01.07	A050-030	001-00034
002-00143	2020.01.07	A020-030	001-00034
002-00144	2020.01.07	A030-130	001-00034
002-00145	2020.01.07	A020-010	001-00040
002-00146	2020.01.07	A030-010	001-00040
002-00147	2020.01.07	A020-020	001-00040
002-00148	2020.01.07	A060-040	001-00040
002-00149	2020.01.07	A040-210	001-00037
002-00150	2020.01.07	A050-010	001-00037
002-00151	2020.01.07	A060-120	001-00037
002-00152	2020.01.07	A030-080	001-00037
002-00153	2020.01.07	A030-140	001-00039
002-00154	2020.01.07	A040-060	001-00039
002-00155	2020.01.07	A050-070	001-00039
002-00156	2020.01.07	A020-090	001-00039
002-00157	2020.01.07	A060-070	001-00038
002-00158	2020.01.07	A050-030	001-00038
002-00159	2020.01.07	A020-030	001-00038
002-00160	2020.01.07	A030-130	001-00038
002-00161	2020.01.07	A020-010	001-00044
002-00162	2020.01.07	A030-010	001-00044
002-00163	2020.01.07	A020-020	001-00044
002-00164	2020.01.07	A060-040	001-00044
002-00165	2020.01.07	A040-210	001-00041
002-00166	2020.01.07	A050-010	001-00041
002-00167	2020.01.07	A060-120	001-00041
002-00168	2020.01.07	A030-080	001-00041
002-00169	2020.01.07	A030-140	001-00043

Anhang 2: Die Datensätze für die Datenbank

002-00170	2020.01.07	A040-060	001-00043
002-00171	2020.01.07	A050-070	001-00043
002-00172	2020.01.07	A020-090	001-00043
002-00173	2020.01.07	A060-070	001-00042
002-00174	2020.01.07	A050-030	001-00042
002-00175	2020.01.07	A020-030	001-00042
002-00176	2020.01.07	A030-130	001-00042
002-00177	2020.01.07	A020-010	001-00048
002-00178	2020.01.07	A030-010	001-00048
002-00179	2020.01.07	A020-020	001-00048
002-00180	2020.01.07	A060-040	001-00048
002-00181	2020.01.07	A040-210	001-00045
002-00182	2020.01.07	A050-010	001-00045
002-00183	2020.01.07	A060-120	001-00045
002-00184	2020.01.07	A030-080	001-00045
002-00185	2020.01.07	A030-140	001-00047
002-00186	2020.01.07	A040-060	001-00047
002-00187	2020.01.07	A050-070	001-00047
002-00188	2020.01.07	A020-090	001-00047
002-00189	2020.01.07	A060-070	001-00046
002-00190	2020.01.07	A050-030	001-00046
002-00191	2020.01.07	A020-030	001-00046
002-00192	2020.01.07	A030-130	001-00046
002-00193	2020.01.09	A020-010	001-00052
002-00194	2020.01.09	A030-010	001-00052
002-00195	2020.01.09	A020-020	001-00052
002-00196	2020.01.09	A060-040	001-00052
002-00197	2020.01.09	A040-210	001-00049
002-00198	2020.01.09	A050-010	001-00049
002-00199	2020.01.09	A060-120	001-00049
002-00200	2020.01.09	A030-080	001-00049
002-00201	2020.01.09	A030-140	001-00051
002-00202	2020.01.09	A040-060	001-00051
002-00203	2020.01.09	A050-070	001-00051
002-00204	2020.01.09	A020-090	001-00051
002-00205	2020.01.09	A060-070	001-00050
002-00206	2020.01.09	A050-030	001-00050
002-00207	2020.01.09	A020-030	001-00050
002-00208	2020.01.09	A030-130	001-00050
002-00209	2020.01.09	A020-010	001-00056
002-00210	2020.01.09	A030-010	001-00056
002-00211	2020.01.09	A020-020	001-00056
002-00212	2020.01.09	A060-040	001-00056
002-00213	2020.01.09	A040-210	001-00053
002-00214	2020.01.09	A050-010	001-00053
002-00215	2020.01.09	A060-120	001-00053

Anhang 2: Die Datensätze für die Datenbank

002-00216	2020.01.09	A030-080	001-00053
002-00217	2020.01.09	A030-140	001-00055
002-00218	2020.01.09	A040-060	001-00055
002-00219	2020.01.09	A050-070	001-00055
002-00220	2020.01.09	A020-090	001-00055
002-00221	2020.01.09	A060-070	001-00054
002-00222	2020.01.09	A050-030	001-00054
002-00223	2020.01.09	A020-030	001-00054
002-00224	2020.01.09	A030-130	001-00054
002-00225	2020.01.09	A020-010	001-00060
002-00226	2020.01.09	A030-010	001-00060
002-00227	2020.01.09	A020-020	001-00060
002-00228	2020.01.09	A060-040	001-00060
002-00229	2020.01.09	A040-210	001-00057
002-00230	2020.01.09	A050-010	001-00057
002-00231	2020.01.09	A060-120	001-00057
002-00232	2020.01.09	A030-080	001-00057
002-00233	2020.01.09	A030-140	001-00059
002-00234	2020.01.09	A040-060	001-00059
002-00235	2020.01.09	A050-070	001-00059
002-00236	2020.01.09	A020-090	001-00059
002-00237	2020.01.09	A060-070	001-00058
002-00238	2020.01.09	A050-030	001-00058
002-00239	2020.01.09	A020-030	001-00058
002-00240	2020.01.09	A030-130	001-00058
002-00241	2020.01.09	A020-010	NULL
002-00242	2020.01.09	A030-010	NULL
002-00243	2020.01.09	A020-020	NULL
002-00244	2020.01.09	A060-040	NULL
002-00245	2020.01.09	A040-210	NULL
002-00246	2020.01.09	A050-010	NULL
002-00247	2020.01.09	A060-120	NULL
002-00248	2020.01.09	A030-080	NULL
002-00249	2020.01.09	A030-140	NULL
002-00250	2020.01.09	A040-060	NULL
002-00251	2020.01.09	A050-070	NULL
002-00252	2020.01.09	A020-090	NULL
002-00253	2020.01.09	A060-070	NULL
002-00254	2020.01.09	A050-030	NULL
002-00255	2020.01.09	A020-030	NULL
002-00256	2020.01.09	A030-130	NULL

Anhang 2: Die Datensätze für die Datenbank

Eingangslager	
PlatzNrEin	TeileNrZulieferer
E-01-A01	NULL
E-01-A02	NULL
E-01-A03	NULL
E-01-A04	002-00242
E-01-B01	NULL
E-01-B02	002-00245
E-01-B03	NULL
E-01-B04	NULL
E-01-C01	NULL
E-01-C02	NULL
E-01-C03	002-00244
E-01-C04	002-00241
E-01-D01	NULL
E-01-D02	002-00243
E-01-D03	NULL
E-01-D04	NULL
E-02-A01	NULL
E-02-A02	002-00247
E-02-A03	002-00248
E-02-A04	NULL
E-02-B01	002-00250
E-02-B02	NULL
E-02-B03	002-00252
E-02-B04	NULL
E-02-C01	NULL
E-02-C02	NULL
E-02-C03	002-00253
E-02-C04	NULL
E-02-D01	NULL
E-02-D02	002-00246
E-02-D03	002-00254
E-02-D04	NULL
E-03-A01	NULL
E-03-A02	002-00255
E-03-A03	NULL
E-03-A04	NULL
E-03-B01	NULL
E-03-B02	002-00256
E-03-B03	NULL
E-03-B04	NULL
E-03-C01	002-00251
E-03-C02	NULL
E-03-C03	002-00249
E-03-C04	NULL
E-03-D01	NULL

Anhang 2: Die Datensätze für die Datenbank

E-03-D02	NULL
E-03-D03	NULL
E-03-D04	NULL
E-04-A01	NULL
E-04-A02	NULL
E-04-A03	NULL
E-04-A04	NULL
E-04-B01	NULL
E-04-B02	NULL
E-04-B03	NULL
E-04-B04	NULL
E-04-C01	NULL
E-04-C02	NULL
E-04-C03	NULL
E-04-C04	NULL
E-04-D01	NULL
E-04-D02	NULL
E-04-D03	NULL
E-04-D04	NULL
E-05-A01	NULL
E-05-A02	NULL
E-05-A03	NULL
E-05-A04	NULL
E-05-B01	NULL
E-05-B02	NULL
E-05-B03	NULL
E-05-B04	NULL
E-05-C01	NULL
E-05-C02	NULL
E-05-C03	NULL
E-05-C04	NULL
E-05-D01	NULL
E-05-D02	NULL
E-05-D03	NULL
E-05-D04	NULL

Anhang 3: Ergebnisse der Lösungswege

Ausgabe durch Lösungsweg 1

PlatzNrEin	TeileNrZulieferer
E-01-C04	002-00241
E-01-A04	002-00242
E-01-D02	002-00243
E-01-C03	002-00244
E-01-B02	002-00245
E-02-D02	002-00246
E-02-A02	002-00247
E-02-A03	002-00248
E-03-C03	002-00249
E-02-B01	002-00250
E-03-C01	002-00251
E-02-B03	002-00252
E-02-C03	002-00253
E-02-D03	002-00254
E-03-A02	002-00255
E-03-B02	002-00256

Ausgabe durch Lösungsweg 2

Anlieferung
2020-01-01
2020-01-03
2020-01-07
2020-01-09

Ausgabe durch Lösungsweg 3

ArtikelNrProdukt	LieferungsNr	Anzahl
A001-029	L780-002	4

Ausgabe durch Lösungsweg 4

RoboterNr	Produktionsdatum	TeileNrProdukt
R085	2020-01-10	001-00060
R085	2020-01-09	001-00052
R085	2020-01-08	001-00044
R085	2020-01-07	001-00036
R085	2020-01-06	001-00028
R085	2020-01-03	001-00020
R085	2020-01-02	001-00012
R085	2020-01-01	001-00004

Ausgabe durch Lösungsweg 5

PlatzNrAus	ArtikelNrProdukt
A-01-A01	A001-176
A-01-B02	A001-176
A-01-C03	A001-176

Ausgabe durch Lösungsweg 6

RoboterNr	HallenNr
R002	H01
R030	H01
R005	H02
R086	H02
R006	H03
R085	H03
R023	H04
R031	H04

⇨

RoboterNr	HallenNr
R002	H01
R030	H01
R001	H02
R005	H02
R086	H02
R006	H03
R085	H03
R023	H04
R031	H04

Anhang 3: Ergebnisse der Lösungswege

Ausgabe durch Lösungsweg 7

RoboterNr	HallenNr
R002	H01
R030	H01
R001	H02
R005	H02
R086	H02
R006	H03
R085	H03
R023	H04
R031	H04

RoboterNr	HallenNr
R005	*NULL*
R002	H01
R030	H01
R001	H02
R086	H02
R006	H03
R085	H03
R023	H04
R031	H04

Ausgabe durch Lösungsweg 8

PlatzNrAus	TeileNrProdukt
A-02-A01	*NULL*
A-02-A02	*NULL*
A-02-A03	*NULL*
A-02-B01	*NULL*
A-02-B02	*NULL*
A-02-B03	*NULL*
A-02-C01	*NULL*
A-02-C02	*NULL*
A-02-C03	*NULL*
A-02-D01	*NULL*
A-02-D02	*NULL*
A-02-D03	*NULL*
A-01-A01	001-00049
A-01-A02	001-00050
A-01-A03	001-00051
A-01-B01	001-00052
A-01-B02	001-00053
A-01-B03	001-00054
A-01-C01	001-00055
A-01-C02	001-00056
A-01-C03	001-00057
A-01-D01	001-00058
A-01-D02	001-00059
A-01-D03	001-00060

PlatzNrAus	TeileNrProdukt
A-02-A01	*NULL*
A-02-A02	*NULL*
A-02-A03	*NULL*
A-02-B01	*NULL*
A-02-B02	*NULL*
A-02-B03	*NULL*
A-02-C01	*NULL*
A-02-C02	*NULL*
A-02-C03	*NULL*
A-02-D01	*NULL*
A-02-D02	*NULL*
A-01-A01	001-00049
A-01-A02	001-00050
A-01-A03	001-00051
A-01-B01	001-00052
A-01-B02	001-00053
A-01-B03	001-00054
A-01-C01	001-00055
A-01-C02	001-00056
A-01-C03	001-00057
A-01-D01	001-00058
A-01-D02	001-00059
A-01-D03	001-00060