DATABASE SYSTEMS
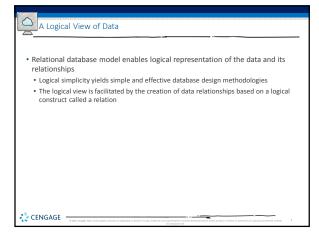DESIGN, IMPLEMENTATION, & MANAGEMENT
CARLOS CORONEL • STEVEN MORRIS
13TH EDITION

Chapter 3
The Relational Database Model

Learning Objectives

- After completing this chapter, you will be able to:
  - Describe the relational database model's logical structure
  - Identify the relational model's basic components and explain the structure, contents, and characteristics of a relational table
  - Use relational database operators to manipulate relational table contents
  - Explain the purpose and components of the data dictionary and system catalog
  - Identify appropriate entities and then the relationships among the entities in the relational database model
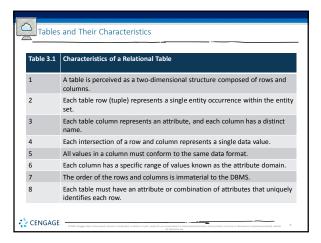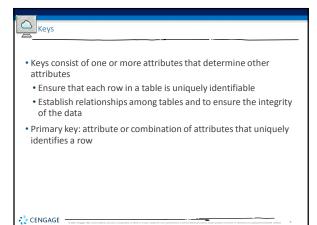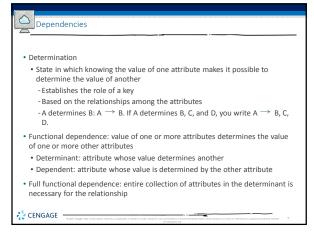
CENGAGE

A Logical View of Data

- Relational database model enables logical representation of the data and its relationships
  - Logical simplicity yields simple and effective database design methodologies
  - The logical view is facilitated by the creation of data relationships based on a logical construct called a relation

CENGAGE

Tables and Their Characteristics

Table 3.1	Characteristics of a Relational Table
1	A table is perceived as a two-dimensional structure composed of rows and columns.
2	Each table row (tuple) represents a single entity occurrence within the entity set.
3	Each table column represents an attribute, and each column has a distinct name.
4	Each intersection of a row and column represents a single data value.
5	All values in a column must conform to the same data format.
6	Each column has a specific range of values known as the attribute domain.
7	The order of the rows and columns is immaterial to the DBMS.
8	Each table must have an attribute or combination of attributes that uniquely identifies each row.

Keys

- Keys consist of one or more attributes that determine other attributes
  - Ensure that each row in a table is uniquely identifiable
  - Establish relationships among tables and to ensure the integrity of the data
- Primary key: attribute or combination of attributes that uniquely identifies a row

Dependencies

- Determination
  - State in which knowing the value of one attribute makes it possible to determine the value of another
    - Establishes the role of a key
    - Based on the relationships among the attributes
    - A determines B: A $\longrightarrow$ B. If A determines B, C, and D, you write A $\longrightarrow$ B, C, D.
- Functional dependence: value of one or more attributes determines the value of one or more other attributes
  - Determinant: attribute whose value determines another
  - Dependent: attribute whose value is determined by the other attribute
- Full functional dependence: entire collection of attributes in the determinant is necessary for the relationship

Student Table

Table name: STUDENT Database name: Ch03_TinyCollege

STU_NUM	STU_LNAME	STU_FNAME	STU_INIT	STU_DOB	STU_HRS	STU_CLASS	STU_GPA	STU_TRANSFER	DEPT_CODE	STU_PHONE	PROF_NUM
321452	Bowser	William	C	12-Feb-1985	42	So	2.84	No	BIOL	2134	205
324257	Smithson	Anne	K	15-Nov-1991	81	Jr	3.27	Yes	CIS	2256	222
324258	Brewer	Juliette		23-Aug-1979	36	So	2.26	Yes	ACCT	2256	228
324269	Oblonski	Walter	H	16-Sep-1986	66	Jr	3.09	No	CIS	2114	222
324273	Smith	John	D	30-Dec-1968	102	Sr	2.11	Yes	ENGL	2231	199
324274	Katinga	Raphael	P	21-Oct-1989	114	Sr	3.15	No	ACCT	2267	228
324291	Robertson	Gerald	T	08-Apr-1983	120	Sr	3.87	No	EDU	2267	311
324299	Smith	John	B	30-Nov-1996	15	Fr	2.92	No	ACCT	2315	230

STU_NUM	= Student number
STU_LNAME	= Student last name
STU_FNAME	= Student first name
STU_INIT	= Student middle initial
STU_DOB	= Student date of birth
STU_HRS	= Credit hours earned
STU_CLASS	= Student classification
STU_GPA	= Grade point average
STU_TRANSFER	= Student transferred from another institution
DEPT_CODE	= Department code
STU_PHONE	= 4-digit campus phone extension
PROF_NUM	= Number of the professor who is the student's advisor

- STU_NUM $\longrightarrow$ STU_LNAME, STU_FNAME, STU_INIT, STU_DOB, STU_TRANSFER

- STU_NUM value determines the STU_LNAME, however, STU_NUM is not functionally dependent on STU_LNAME. Why?

Types of Keys (1 of 3)

- Several different types of keys are used in the relational model
  - Composite key: key that is composed of more than one attribute
  - Key attribute: attribute that is a part of a key
    - STU_LNAME, STU_FNAME, STU_INIT, STU_PHONE $\longrightarrow$ STU_HRS, STU_GPA

  - Superkey: key that can uniquely identify any row in the table
    - STU_NUM
  - Candidate key: minimal superkey (i.e. a superkey without redundancies)
    - STU_LNAME, STU_FNAME, STU_INIT, STU_PHONE
    - STU_NUM, STU_LNAME cannot be a candidate key because STU_NUM by itself is a superkey

  - Foreign key: primary key of one table that has been placed into another table to create a common attribute
  - Secondary key: key used strictly for data retrieval purposes

---

Types of Keys

- Entity integrity: condition in which each row in the table has its own unique identity
  - All of the values in the primary key must be unique to ensure that each row is uniquely identified
  - No key attribute in the primary key can contain a null
- Null: absence of any data value
  - Unknown attribute value, known but missing attribute value, or inapplicable condition
- Referential integrity: if the foreign key contains a value, that value refers to an existing valid tuple (row) in another relation. See fig. 3.2 below

FIGURE 3.2 AN EXAMPLE OF A SIMPLE RELATIONAL DATABASE

Table name: PRODUCT
Primary key: PROD_CODE
Foreign key: VEND_CODE

Database name: Ch03_SaleCo

PROD_CODE	PROD_DESCRIPT	PROD_PRICE	PROD_ON_HAND	VEND_CODE
001278-AB	Claw hammer	12.95	23	232
123-21UUY	Houselite chain saw, 16-in. bar	189.99	4	235
QER-34256	Sledge hammer, 16-lb. head	18.63	6	231
SRE-657UG	Rat-tail file	2.99	15	232
ZZX/3245Q	Steel tape, 12-ft. length	6.79	8	235

link

Table name: VENDOR
Primary key: VEND_CODE
Foreign key: none

VEND_CODE	VEND_CONTACT	VEND_AREACODE	VEND_PHONE
230	Shelly K. Smithson	608	555-1234
231	James Johnson	615	123-4536
232	Annelise Crystall	608	224-2134
233	Candice Wallace	904	342-6567
234	Arthur Jones	615	123-3324
235	Henry Ortozo	615	899-3425

CENGAGE

Table 3.3	Relational Database Keys
Key Type	**Definition**
Superkey	An attribute or combination of attributes that uniquely identifies each row in a table.
Candidate key	A minimal (irreducible) superkey; a superkey that does not contain a subset of attributes that is itself a superkey.
Primary key	A candidate key selected to uniquely identify all other attribute values in any given row; cannot contain null entries.
Foreign key	An attribute or combination of attributes in one table whose values must either match the primary key in another table or be null.
Secondary key	An attribute or combination of attributes used strictly for data retrieval purposes.

CENGAGE

Relational Schema

- A Relational Schema is a textual representation of the database tables where each table is described by its name followed by the list of its attributes in parenthesis.

- CUSTOMER (**CUS_CODE**, CUS_NAME, CUS_FNAME, CUS_INITIAL, CUS_AREACODE, CUS_PHONE, CUS_INSURE_TYPE, AGENT_CODE)

AGENT
- AGENT_CODE
- AGENT_LNAME
- AGENT_FNAME
- AGENT_INITIAL
- AGENT_AREACODE
- AGENT_PHONE
- AGENT_ADDRESS
- AGENT_CITY
- AGENT_STATE
- AGENT_ZIP
- AGENT_DATE_HIRED
- AGENT_YTD_PAY

CUSTOMER
- CUS_CODE
- CUS_LNAME
- CUS_FNAME
- CUS_INITIAL
- CUS_AREACODE
- CUS_PHONE
- CUS_INSURE_TYPE
- CUS_INSURE_AMT
- CUS_RENEW_DATE
- AGENT_CODE

CENGAGE

4

- Relational database integrity rules are very important to good database design
  - Relational database management systems (RDBMSs) enforce integrity rules automatically
    - Much safer to make sure the application design conforms to entity and referential integrity rules

Table 3.4	Integrity Rules
Entity Integrity	**Description**
Requirement	All primary key entries are unique, and no part of a primary key may be null.
Purpose	Each row will have a unique identity, and foreign key values can properly reference primary key values.
Example	No invoice can have a duplicate number, nor can it be null; in short, all invoices are uniquely identified by their invoice number.
Referential Integrity	**Description**
Requirement	A foreign key may have either a null entry, as long as it is not a part of its table's primary key, or an entry that matches the primary key value in a table to which it is related (every non-null foreign key value must reference an existing primary key value).
Purpose	It is possible for an attribute not to have a corresponding value, but it will be impossible to have an invalid entry; the enforcement of the referential integrity rule makes it impossible to delete a row in one table whose primary key has mandatory matching foreign key values in another table.
Example	A customer might not yet have an assigned sales representative (number), but it will be impossible to have an invalid sales representative (number).

FIGURE 3.3 AN ILLUSTRATION OF INTEGRITY RULES

Table name: CUSTOMER   Database name: Ch03_InsureCo
Primary key: CUS_CODE
Foreign key: AGENT_CODE

CUS_CODE	CUS_LNAME	CUS_FNAME	CUS_INITIAL	CUS_RENEW_DATE	AGENT_CODE
10010	Ramas	Alfred	A	05-Apr-2018	502
10011	Dunne	Leona	K	16-Jun-2018	501
10012	Smith	Kathy	W	29-Jan-2019	502
10013	Olowski	Paul	F	14-Oct-2018	
10014	Orlando	Myron		28-Dec-2018	501
10015	O'Brian	Amy	B	22-Sep-2018	503
10016	Brown	James	G	25-Mar-2019	502
10017	Williams	George		17-Jul-2018	503
10018	Farriss	Anne	G	03-Dec-2018	501
10019	Smith	Olette	K	14-Mar-2019	503

Table name: AGENT (only five selected fields are shown)
Primary key: AGENT_CODE
Foreign key: none

AGENT_CODE	AGENT_AREACODE	AGENT_PHONE	AGENT_LNAME	AGENT_YTD_SLS
501	713	228-1249	Alby	132735.75
502	615	882-1244	Hahn	138967.35
503	615	123-5589	Okon	127093.45

- Entity Integrity: CUSTOMER table primary key is CUS_CODE. The primary key column has no null entries, and all entries are unique. Similarly, AGENT table has AGENT_CODE as primary key and the column is free of null entries.

- Referential integrity: AGENT_CODE entries in CUSTOMER table all match the entries in AGENT table.

- Ways to handle nulls
  - Flags
    - Special codes used to indicate the absence of some value. A "-99" value can be inserted to indicate that nothing has been assigned to that row yet.
  - Constraints
    - NOT NULL constraint: placed on a column to ensure that every row in the table has a value for that column
    - UNIQUE constraint: restriction placed on a column to ensure that no duplicate values exist for that column

CENGAGE
© 2020 Cengage. May not be copied, scanned, or duplicated, in whole or in part, except for use as permitted in a license distributed with a certain product or service or otherwise on a password-protected website for classroom use. 16

Test your knowledge (teamwork)

Database name: Ch03_BeneCo

Table name: EMPLOYEE

EMP_CODE	EMP_LNAME	JOB_CODE
14	Rudell	2
15	McDade	1
16	Ruellardo	1
17	Smith	3
20	Smith	2

Table name: BENEFIT

EMP_CODE	PLAN_CODE
15	2
15	3
16	1
17	1
17	3
17	4
20	3

Table name: JOB

JOB_CODE	JOB_DESCRIPTION
1	Clerical
2	Technical
3	Managerial

Table name: PLAN

PLAN_CODE	PLAN_DESCRIPTION
1	Term life
2	Stock purchase
3	Long-term disability
4	Dental

A) For each table in the database, identify the primary key and the foreign key(s).

If a table does not have a foreign key, write None.

B) Do the tables exhibit entity and referential integrity? Yes/No then explain your answer

CENGAGE
© 2020 Cengage. May not be copied, scanned, or duplicated, in whole or in part, except for use as permitted in a license distributed with a certain product or service or otherwise on a password-protected website for classroom use. 17

Solution

TABLE	ENTITY INTEGRITY	EXPLANATION	REFERENTIAL INTEGRITY	EXPLANATION
EMPLOYEE	Yes	Each EMP_CODE value is unique and there are no nulls.	Yes	Each JOB_CODE value in EMPLOYEE points to an existing JOB_CODE value in JOB.
BENEFIT	Yes	Each combination of EMP_CODE and PLAN_CODE values is unique and there are no nulls.	Yes	Each EMP_CODE value in BENEFIT points to an existing EMP_CODE value in EMPLOYEE and each PLAN_CODE value in BENEFIT points to an existing PLAN_CODE value in PLAN.
JOB	Yes	Each JOB_CODE value is unique and there are no nulls.	N/A	
PLAN	Yes	Each PLAN_CODE value is unique and there are no nulls.	N/A	

6

- Select (restrict)
  - Unary operator that yields a horizontal subset of a table
- Project
  - Unary operator that yields a vertical subset of a table
- Union
  - Combines all rows from two tables, excluding duplicate rows
  - Union-compatible: tables share the same number of columns, and their corresponding columns share compatible domains
- Intersect
  - Yields only the rows that appear in both tables
  - Tables must be union-compatible to yield valid results

FIGURE 3.4 SELECT

FIGURE 3.5 PROJECT

FIGURE 3.6 UNION

FIGURE 3.7 INTERSECT

- Difference
  - Yields all rows in one table that are not found in the other table
  - Tables must be union-compatible to yield valid results
- Product
  - Yields all possible pairs of rows from two tables

FIGURE 3.8 DIFFERENCE

FIGURE 3.9 PRODUCT

8

- Joins allow information to be intelligently combined from two or more tables
  - Natural join: links tables by selecting only the rows with common values in their common attribute
  - Inner join: only returns matched records from the tables that are being joined
  - Outer join: matched pairs are retained and unmatched values in the other table are left null
    - Left outer join: yields all of the rows in the first table, including those that do not have a matching value in the second table
    - Right outer join: yields all of the rows in the second table, including those that do not have matching values in the first table

---

- Divide
  - Uses one double-column table as the dividend and one single-column table as the divisor
  - Output is a single column that contains all values from the second column of the dividend that are associated with every row in the divisor

---

FIGURE 3.10 TWO TABLES THAT WILL BE USED IN JOIN ILLUSTRATIONS

Table name: CUSTOMER

CUS_CODE	CUS_LNAME	CUS_ZIP	AGENT_CODE
1132445	Walker	32145	231
1217782	Adares	32145	125
1312243	Rakowski	34129	167
1321242	Rodriguez	37134	125
1542311	Smithson	37134	421
1657399	Vanloo	32145	231

Table name: AGENT

AGENT_CODE	AGENT_PHONE
125	6152439887
167	6153426778
231	6152431124
333	9041234445

Step 1 – PRODUCT of the Table is created

- A natural join links tables by selecting only the rows with common values in their common attributes.

FIGURE 3.11 NATURAL JOIN, STEP 1: PRODUCT

CUS_CODE	CUS_LNAME	CUS_ZIP	CUSTOMER.AGENT_CODE	AGENT.AGENT_CODE	AGENT_PHONE
1132445	Walker	32145	231	125	6152439887
1132445	Walker	32145	231	167	6153426778
1132445	Walker	32145	231	231	6152431124
1132445	Walker	32145	231	333	9041234445
1217782	Adares	32145	125	125	6152439887
1217782	Adares	32145	125	167	6153426778
1217782	Adares	32145	125	231	6152431124
1217782	Adares	32145	125	333	9041234445
1312243	Rakowski	34129	167	125	6152439887
1312243	Rakowski	34129	167	167	6153426778
1312243	Rakowski	34129	167	231	6152431124
1312243	Rakowski	34129	167	333	9041234445
1321242	Rodriguez	37134	125	125	6152439887
1321242	Rodriguez	37134	125	167	6153426778
1321242	Rodriguez	37134	125	231	6152431124
1321242	Rodriguez	37134	125	333	9041234445
1542311	Smithson	37134	421	125	6152439887
1542311	Smithson	37134	421	167	6153426778
1542311	Smithson	37134	421	231	6152431124
1542311	Smithson	37134	421	333	9041234445
1657399	Vanloo	32145	231	125	6152439887
1657399	Vanloo	32145	231	167	6153426778
1657399	Vanloo	32145	231	231	6152431124
1657399	Vanloo	32145	231	333	9041234445

CENGAGE

Step 2: SELECT

- This yields only the rows for which AGENT_CODE values are equal

FIGURE 3.12 NATURAL JOIN, STEP 2: SELECT

CUS_CODE	CUS_LNAME	CUS_ZIP	CUSTOMER.AGENT_CODE	AGENT.AGENT_CODE	AGENT_PHONE
1217782	Adares	32145	125	125	6152439887
1321242	Rodriguez	37134	125	125	6152439887
1312243	Rakowski	34129	167	167	6153426778
1132445	Walker	32145	231	231	6152431124
1657399	Vanloo	32145	231	231	6152431124

CENGAGE

Step 3: PROJECT

- This is performed on the results of Step 2 to yield a single copy of each attribute, thereby eliminating duplicate columns.

FIGURE 3.13 NATURAL JOIN, STEP 3: PROJECT

CUS_CODE	CUS_LNAME	CUS_ZIP	AGENT_CODE	AGENT_PHONE
1217782	Adares	32145	125	6152439887
1321242	Rodriguez	37134	125	6152439887
1312243	Rakowski	34129	167	6153426778
1132445	Walker	32145	231	6152431124
1657399	Vanloo	32145	231	6152431124

- The column AGENT_CODE on which the join was made occurs only once in the new table.

CENGAGE

10

Left Outer Join

- Left Outer Join: Yields all of the rows in the CUSTORMER table, including those that do not have a matching value in he AGENT table.

FIGURE 3.14 LEFT OUTER JOIN

CUS_CODE	CUS_LNAME	CUS_ZIP	CUSTOMER.AGENT_CODE	AGENT.AGENT_CODE	AGENT_PHONE
1217782	Adares	32145	125	125	6152439887
1321242	Rodriguez	37134	125	125	6152439887
1312243	Rakowski	34129	167	167	6153426778
1132445	Walker	32145	231	231	6152431124
1657399	Vanloo	32145	231	231	6152431124
1542311	Smithson	37134	421		

- Right Outer Join: Yields all of the rows in the AGENT table, including those that do not have a matching value in he CUSTOMER table.

FIGURE 3.15 RIGHT OUTER JOIN

CUS_CODE	CUS_LNAME	CUS_ZIP	CUSTOMER.AGENT_CODE	AGENT.AGENT_CODE	AGENT_PHONE
1217782	Adares	32145	125	125	6152439887
1321242	Rodriguez	37134	125	125	6152439887
1312243	Rakowski	34129	167	167	6153426778
1132445	Walker	32145	231	231	6152431124
1657399	Vanloo	32145	231	231	6152431124
				333	9041234445

Test your knowledge

Table name: STUDENT

STU_CODE	PROF_CODE
100278	
128569	2
512272	4
531235	2
531268	
553427	1

Table name: PROFESSOR

PROF_CODE	DEPT_CODE
1	2
2	6
3	6
4	4

- Given the STUDENT and PROFESSOR tables,

A) illustrate how the JOINS will be presented

I.   Natural Join (3 Steps – PRODUCT, SELECT, PROJECT)

II.   Left Outer Join of STUDENT to PROFESSOR

III. Right Outer Join of STUDENT to PROFESSOR

B) Create the Relationship Diagram

Solution

STU_CODE	STUDENT. PROF_CODE	PROFESSOR. PROF_CODE	DEPT_CODE
128569	2	2	6
512272	4	4	4
531235	2	2	6
553427	1	1	2

STU_CODE	PROF_CODE	DEPT_CODE
128569	2	6
512272	4	4
531235	2	6
553427	1	2

Solution

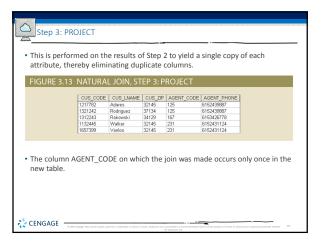- Left Outer Join of STUDENT to PROFESSOR

STU_CODE	STUDENT. PROF_CODE	PROFESSOR. PROF_CODE	DEPT_CODE
128569	2	2	6
512272	4	4	4
531235	2	2	6
553427	1	1	2
100278			
531268			

- Right Outer Join of STUDENT to PROFESSOR

STU_CODE	STUDENT. PROF_CODE	PROFESSOR. PROF_CODE	DEPT_CODE
128569	2	2	6
512272	4	4	4
531235	2	2	6
553427	1	1	2
		3	6

Data Dictionary and the System Catalog

- Data dictionary
  - Description of all tables in the database created by the user and designer
- System catalog
  - System data dictionary that describes all objects within the database
- Homonyms and synonyms must be avoided to lessen confusion
  - Homonym: same name is used to label different attributes
  - Synonym: different names are used to describe the same attribute

---

Relationships within the Relational Database (1 of 4)

- One-to-many (1:M)
  - Norm for relational databases
- One-to-one (1:1)
  - One entity can be related to only one other entity and vice versa
- Many-to-many (M:N)
  - Implemented by creating a new entity in 1:M relationships with the original entities
  - Composite entity (i.e., bridge or associative entity): helps avoid problems inherent to M:N relationships
    - Includes the primary keys of tables to be linked

Relationships within the Relational Database

FIGURE 3.22 THE IMPLEMENTED 1:1 RELATIONSHIP BETWEEN PROFESSOR AND DEPARTMENT

1:1 Relationship between professor and department

FIGURE 3.21 THE 1:1 RELATIONSHIP BETWEEN PROFESSOR AND DEPARTMENT

M:N Relationship between STUDENT and CLASS

FIGURE 3.24 THE WRONG IMPLEMENTATION OF THE M:N RELATIONSHIP BETWEEN STUDENT AND CLASS

FIGURE 3.23 THE ERM'S M:N RELATIONSHIP BETWEEN STUDENT AND CLASS

Slide 1

Table name: STUDENT
Primary key: STU_NUM
Foreign key: none

Database name: Ch03_CollegeTry2

STU_NUM	STU_LNAME
321452	Bowser
324257	Smithson

Table name: ENROLL
Primary key: CLASS_CODE + STU_NUM
Foreign key: CLASS_CODE, STU_NUM

CLASS_CODE	STU_NUM	ENROLL_GRADE
10014	321452	C
10014	324257	B
10018	321452	A
10018	324257	B
10021	321452	C
10021	324257	C

Table name: CLASS
Primary key: CLASS_CODE
Foreign key: CRS_CODE

CLASS_CODE	CRS_CODE	CLASS_SECTION	CLASS_TIME	CLASS_ROOM	PROF_NUM
10014	ACCT-211	3	TTh 2:30-3:45 p.m.	BUS252	342
10018	CIS-220	2	MWF 9:00-9:50 a.m.	KLR211	114
10021	QM-261	1	MWF 8:00-8:50 a.m.	KLR200	114

CENGAGE

Slide 2

Relationships within the Relational Database

CENGAGE

Slide 3

Test your knowledge

During some time interval, a DRIVER can drive many TRUCKs
and any TRUCK can be driven by many DRIVERs

- How would you convert this M:N model into an ER that displays only 1:M relationships?

(hint: introduce "Assignments" for each driver and write the business rules before drawing the new ERD)

CENGAGE

Solution

- A driver may receive many (driving) assignments.
- Each (driving) assignment is made for a single driver.
- A truck may be driven in many (driving) assignments.
- Each (driving) assignment is made for a single truck.

A driver can drive only one truck at a time, but during some period of time, a driver may be assigned to drive many trucks. The same argument holds true for trucks – a truck can only be driven during one trip (assignment) at a time, but during some period of time, a truck may be assigned to be driven in many trips.