STUDENT LEARNING OUTCOMES

1. Identify how databases and spreadsheets are both similar and different.
2. List and describe the four steps in designing and building a relational database.
3. Define the concepts of entity class, instance, primary key, and foreign key.
STUDENT LEARNING OUTCOMES

4. Given a small operating environment, build an entity-relationship diagram.
5. List and describe the steps in normalization.
6. Describe the process of creating an intersection relation to remove a many-to-many relationship.

INTRODUCTION

- Chapter 3 discussed why databases are important
- This module teaches you how to design a relational database
- Relational databases are the most popular model
INTRODUCTION

- Databases and spreadsheets are similar and different
- Both have rows and columns of information
  - Spreadsheet – must know physical row and column (e.g., B4)
  - Database – work with information logically

- **Database** – collection of information that you organize and access according to the logical structure of that information
- **Relational database** – uses a series of logically related two-dimensional tables or files to store information in the form of a database
DESIGNING & BUILDING A RELATIONAL DATABASE

1. Define entity classes & primary keys
2. Define relationships among classes
3. Define information for each relation
   - Relation = table = file
4. Use a data definition language to create database

Remember Solomon Enterprises?

- From Chapter 3
- Provides concrete to commercial builders & home owners
- Chapter 3 – the CRM side of Solomon’s database
- Focus now – SCM side of Solomon’s database
Remember Solomon Enterprises?

- Customer’s database tables
  - Customer
  - Concrete Type
  - Order
  - Truck
  - Employee

### Customer File

<table>
<thead>
<tr>
<th>Customer ID</th>
<th>Customer Name</th>
<th>Contact Person</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>12395</td>
<td>Namey Co.</td>
<td>John</td>
<td>555-1234</td>
</tr>
<tr>
<td>56789</td>
<td>Smith &amp; Jones</td>
<td>James</td>
<td>555-5432</td>
</tr>
<tr>
<td>34567</td>
<td>Lee Enterprises</td>
<td>Robert</td>
<td>555-6789</td>
</tr>
</tbody>
</table>

### Concrete Type File

<table>
<thead>
<tr>
<th>Concrete Type</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Ready-mixed concrete with sand aggregate</td>
<td>Concrete Type ID: 1234</td>
</tr>
<tr>
<td>2</td>
<td>Pre-cast concrete with sand aggregate</td>
<td>Concrete Type ID: 5678</td>
</tr>
<tr>
<td>3</td>
<td>Concrete with sand aggregate</td>
<td>Concrete Type ID: 1234</td>
</tr>
<tr>
<td>4</td>
<td>Ready-mixed concrete with crushed stone aggregate</td>
<td>Concrete Type ID: 5678</td>
</tr>
<tr>
<td>5</td>
<td>Pre-cast concrete with crushed stone aggregate</td>
<td>Concrete Type ID: 1234</td>
</tr>
</tbody>
</table>

### Order File

<table>
<thead>
<tr>
<th>Order No.</th>
<th>Date</th>
<th>Customer ID</th>
<th>Concrete Type</th>
<th>Truck ID</th>
<th>Employee ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>12365</td>
<td>1/1/2022</td>
<td>12395</td>
<td>1</td>
<td>111</td>
<td>34567</td>
</tr>
<tr>
<td>78901</td>
<td>2/2/2022</td>
<td>56789</td>
<td>2</td>
<td>222</td>
<td>34567</td>
</tr>
<tr>
<td>23456</td>
<td>3/3/2022</td>
<td>34567</td>
<td>3</td>
<td>333</td>
<td>34567</td>
</tr>
</tbody>
</table>

### Truck File

<table>
<thead>
<tr>
<th>Truck ID</th>
<th>Brand</th>
<th>Model</th>
<th>Date of Purchase</th>
</tr>
</thead>
<tbody>
<tr>
<td>111</td>
<td>Ford</td>
<td>12/20/2011</td>
<td></td>
</tr>
<tr>
<td>222</td>
<td>Ford</td>
<td>12/24/2011</td>
<td></td>
</tr>
<tr>
<td>333</td>
<td>Chevy</td>
<td>1/1/2012</td>
<td></td>
</tr>
</tbody>
</table>

### Employee File

<table>
<thead>
<tr>
<th>Employee ID</th>
<th>First Name</th>
<th>Last Name</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>123456789</td>
<td>John</td>
<td>Smith</td>
<td>555-1234</td>
</tr>
<tr>
<td>456789012</td>
<td>Jane</td>
<td>Lee</td>
<td>555-5678</td>
</tr>
<tr>
<td>234567890</td>
<td>Bob</td>
<td>Johnson</td>
<td>555-7890</td>
</tr>
</tbody>
</table>

### Solomon Enterprises’ database tables

- **Customer**
- **Concrete Type**
- **Order**
- **Truck**
- **Employee**
Observations for Solomon

- 5 concrete types
- 1. Home foundation and walkways
- 2. Commercial foundation and infrastructure
- 3. Premier speckled (with gravel)
- 4. Premier marble
- 5. Premier shell
Observations for Solomon

- 6 raw materials
  A. Water
  B. Cement paste
  C. Sand
  D. Gravel
  E. Marble
  F. Shell
- Mixing instructions are for a cubic yard

Some raw materials are in several concrete types
Concrete types require several raw materials
Inventory (QOH) is tracked for all raw materials
Observations for Solomon

- Suppliers provide raw materials
- Solomon uses only 1 supplier for a given raw material
- A supplier can provide several raw materials

Observations for Solomon

- Water
  - Supplier not tracked
  - QOH not tracked
Business Rules for Solomon

1. Given concrete type will have many raw materials
2. Given raw material may appear in many concrete types
3. Each raw material has one and only one supplier

4. A supplier may provide many raw materials
   - There may be suppliers present not providing any raw materials
   - These business rules are very important to remember
STEP 1: DEFINE ENTITY CLASSES & PRIMARY KEYS

- **Entity class** – concept – typically, person, place, or thing – about which you wish to store information and that you can identify with a unique key (primary key)
  - Concrete Type
  - Raw Material
  - Supplier

- **Primary key** – a field (or group of fields) that uniquely describe each record
- A record in a database is sometimes called an **instance** (of an entity class)
STEP 1: DEFINE ENTITY CLASSES & PRIMARY KEYS

- In general, stay away from names for primary keys (duplicates)
- **Concrete Type** – **Concrete Type**
- **Raw Material** – **Raw Material ID**
- **Supplier** – **Supplier ID**

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>ID</th>
<th>Name</th>
<th>Unit</th>
<th>QOH</th>
<th>ID</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Water</td>
<td>1</td>
<td>Cement paste</td>
<td>1</td>
<td>400</td>
<td>452</td>
<td>Whiskey Enterprises</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Sand</td>
<td>2</td>
<td>1200</td>
<td>448</td>
<td>Jupiter Sand &amp; Gravel</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>A</td>
<td>Water</td>
<td>1.5</td>
<td>9099</td>
<td>916</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TOTAL</td>
<td></td>
<td></td>
<td></td>
<td>4.5</td>
</tr>
<tr>
<td>2</td>
<td>Cement paste</td>
<td>1</td>
<td>400</td>
<td>452</td>
<td>Whiskey Enterprises</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Sand</td>
<td>2</td>
<td>1200</td>
<td>448</td>
<td>Jupiter Sand &amp; Gravel</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>A</td>
<td>Water</td>
<td>1</td>
<td>9099</td>
<td>916</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TOTAL</td>
<td></td>
<td></td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>Specified</td>
<td>1</td>
<td>400</td>
<td>452</td>
<td>Whiskey Enterprises</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Sand</td>
<td>2</td>
<td>1200</td>
<td>448</td>
<td>Jupiter Sand &amp; Gravel</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>A</td>
<td>Water</td>
<td>1.5</td>
<td>9099</td>
<td>916</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>D</td>
<td>Gravel</td>
<td>3</td>
<td>200</td>
<td>444</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TOTAL</td>
<td></td>
<td></td>
<td></td>
<td>7.5</td>
</tr>
<tr>
<td>4</td>
<td>Marble</td>
<td>1</td>
<td>Cement paste</td>
<td>1</td>
<td>400</td>
<td>452</td>
<td>Whiskey Enterprises</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Sand</td>
<td>2</td>
<td>1200</td>
<td>448</td>
<td>Jupiter Sand &amp; Gravel</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>A</td>
<td>Water</td>
<td>1.5</td>
<td>9099</td>
<td>916</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>E</td>
<td>Marble</td>
<td>2</td>
<td>100</td>
<td>409</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TOTAL</td>
<td></td>
<td></td>
<td></td>
<td>3.3</td>
</tr>
<tr>
<td>5</td>
<td>Shale</td>
<td>1</td>
<td>Cement paste</td>
<td>1</td>
<td>400</td>
<td>452</td>
<td>Whiskey Enterprises</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Sand</td>
<td>2</td>
<td>1200</td>
<td>448</td>
<td>Jupiter Sand &amp; Gravel</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>A</td>
<td>Water</td>
<td>1.5</td>
<td>9099</td>
<td>916</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>Shale</td>
<td>2.5</td>
<td>25</td>
<td>409</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TOTAL</td>
<td></td>
<td></td>
<td></td>
<td>3</td>
</tr>
</tbody>
</table>
STEP 2: DEFINE RELATIONSHIPS AMONG ENTITY CLASSES

- To define relationships, you create an E-R diagram
- **Entity-relationship (E-R) diagram** – a graphic method of representing entity classes and their relationships

E-R Diagrams

- Use 5 symbols
  1. Rectangle – entity class
  2. Dotted line – relationship
  3. | - single relationship
  4. 0 – zero/optional relationship
  5. Crow’s foot (←) – multiple relationship
Figure C.2 on page 147 reads as…

- A Concrete Type is composed of Raw Material
- A Raw Material is used to create a Concrete Type
- A Supplier provides a Raw Material
- A Raw Material is provided by a Supplier
E-R Diagrams - Cardinality

- E-R diagrams show relationships.
- They also show numerical nature of relationships.
- This is called cardinality.
  - | - single relationship
  - 0 – zero/optional relationship
  - ← - multiple relationship

Following lines marked A:
- A Supplier may not provide any Raw Material (0) but may provide more than one Raw Material (←)
E-R Diagrams - Cardinality

- Following lines marked B:
  - A Raw Material must be provided by a Supplier (\(|\)) and can only be provided by one Supplier (\(|\))

Normalization

- **Normalization** – process of assuring that a relational database structure can be implemented as a series of two-dimensional tables
- We will follow three rules of normalization
Normalization Rules

1. Eliminate repeating groups or many-to-many relationships
2. Assure that each field in a relation depends only on the primary key for that relation
3. Remove all derived fields from the relations

Eliminating Many-to-Many Relationships

- A many-to-many relationship exists if there is a crow’s foot (≡) on each end
- You must eliminate these by creating an intersection relation
Eliminating Many-to-Many Relationships

- **Intersection (composite) relation** – a relation you create to eliminate a many-to-many relationship
- Intersection relation will have a composite primary key
- **Composite primary key** – consists of the primary key fields from the two intersecting relations

<table>
<thead>
<tr>
<th>CONCRETE TYPE</th>
<th>BILL OF MATERIAL</th>
<th>RAW MATERIAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary Keys</td>
<td>Primary Keys</td>
<td>Primary Keys</td>
</tr>
<tr>
<td>1</td>
<td>1 + B</td>
<td>A</td>
</tr>
<tr>
<td>2</td>
<td>2 + B</td>
<td>B</td>
</tr>
<tr>
<td>3</td>
<td>1 + C</td>
<td>C</td>
</tr>
<tr>
<td>4</td>
<td>2 + C</td>
<td>D</td>
</tr>
<tr>
<td>5</td>
<td>3 + B</td>
<td>E</td>
</tr>
<tr>
<td></td>
<td>4 + B</td>
<td>F</td>
</tr>
<tr>
<td></td>
<td>3 + C</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4 + C</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 + A</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4 + A</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 + D</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4 + E</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5 + B</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5 + C</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5 + A</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5 + F</td>
<td></td>
</tr>
</tbody>
</table>
Steps to Eliminate a Many-to-Many Relationship

1. Draw the part of the E-R diagram with many-to-many relationship
2. Write some primary keys for each relation
3. Create new E-R diagram with intersection relation in the middle
4. Write some composite primary keys for intersection relation
Steps to Eliminate a Many-to-Many Relationship

5. Create a meaningful name for intersection relation
6. Move minimum cardinality next to left relation to the right of intersection relation
7. Move minimum cardinality next to right relation to the left of intersection relation

8. Maximum cardinality on both sides of intersection relation is always many (\(\geq\))
9. General rule – new minimum and maximum cardinalities for the 2 original relations will be one (\(1\)) and one (\(1\))
STEP 3: DEFINING INFORMATION FOR EACH RELATION

- To ensure that each field is in the right relation, ask the following question:
- “Does this piece of information depend only on the primary key for this relation?”
  - Yes – it’s in the correct relation
  - No – It’s in the wrong relation

See Figure C.6 on page 153
- Look at Raw Material relation
- Every field must depend only on Raw Material ID
- Raw Material Name, QOH, and Supplier ID do
- Supplier Name does not
STEP 3: DEFINING INFORMATION FOR EACH RELATION

- What does Supplier Name depend on?
- It depends on Supplier ID
- Supplier ID is primary key for Supplier relation
- Therefore, Supplier Name belongs only in Supplier relation

STEP 3: DEFINING INFORMATION FOR EACH RELATION

- See Figure C.6 on page 153
- Do you see any derived information?
  - Counts?
  - Sums?
  - Averages?
- If you see them, remove them
STEP 3: DEFINING INFORMATION FOR EACH RELATION

- *Raw Material Total* in the *Concrete Type* relation is derived.
- It can be obtained by summing the appropriate fields of the appropriate records in the *Bill of Material* relation.
- Therefore, you do not need *Raw Material Total*.
- Take it out.

---

STEP 3: DEFINING INFORMATION FOR EACH RELATION

- See Figure C.7 on page 154.
- It is the correct and final structure.
- No many-to-many relationships.
- Each field depends only on relation’s primary key.
- No derived fields.
- Good database design.
STEP 4: USE A DATA DEFINITION LANGUAGE TO CREATE YOUR DATABASE

- You’re ready to implement Solomon’s database with a DBMS
- *Database management system (DBMS)* – helps you specify the logical organization for a database and access and use the information within the database

STEP 4: USE A DATA DEFINITION LANGUAGE TO CREATE YOUR DATABASE

- When creating a database, you must first create the data dictionary
- *Data dictionary* – contains the logical structure for the information in a database
- This is the first step in implementing your database
- *Extended Learning Module J* is devoted to using Microsoft Access to create Solomon’s database