Relational Databases, Entity Relationship Modeling

CS 360 Internet Programming

Daniel Zappala

Brigham Young University
Computer Science Department

Tables

- each database contains a collection of tables
  - each row is a unique record
  - each column is an attribute of the record

Juicery Table

Juicery ID	Juicery Name	Address	Region ID
1	Moss Brothers	Smith Rd.	3
2	Hardy Brothers	Jones St.	1
3	Penfolds	Arthurton Rd.	1
4	Lindemans	Smith Ave.	2
5	Orlando	Jones St.	1

Region Table

Region ID	Region Name	State
1	Barossa Valley	South Australia
2	Yarra Valley	Victoria
3	Margaret River	Western Australia

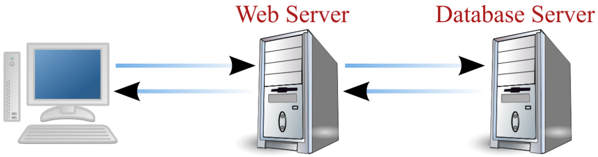
Attributes

- attributes have data types
  - Juicery and Region IDs are integers
  - Juicery Name and Address are strings
- primary key
  - one or more keys that together uniquely identify each row in a table
  - Juicery ID and Region ID in our example

Relationships

- form relationships between tables using identifiers
    - a juicery has a particular location; could create attributes for the region name, state in the juicery table
    - but there is a one-to-many mapping between regions and juiceries, so much of this information will be redundant
    - use unique identifier for the region in the juicery table
- may also have one-to-one and many-to-many relationships
- must have unique identifiers

Three-Tier Architecture

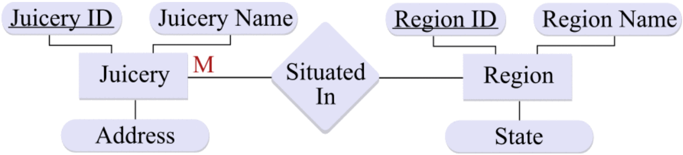


- database server (DBMS)
  - manages a set of databases
  - supports a query language: SQL
  - accessed via a database driver
- web and database servers may run on the same machine or different machines
  - even with a single machine, can handle $10,000+$ requests per hour
  - for higher loads, distribute load for both servers across a cluster of machines

Advantages of a Web Database Server

- supports multiple clients at a time
  - very few applications are written for one user
  - provides concurrency and security
- the power of the web
  - standardized and widely supported
  - any user can access the database using any OS and browser

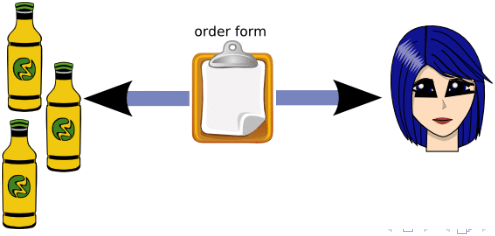
Entity-Relationship Model



- visualizes database and its relationships
  - tables (entities): rectangles
  - attributes: ellipses
  - relationship: diamond
- primary key is underlined
- relationship is annotated with an M, showing a one-to-many relationship

Entities and Relationships

- juice store example: three main entities
  - juices
  - customer
  - order
- purchasing action creates a relationship between customer, order, and juice
  - must associate one customer with each order
  - customers can make more than one order
  - each order has one or more bottles of juice



Identifying Entities

- entity: objects that can be described by characteristics
  - juice: juice name, description, year
  - customer: surname, firstname, initial, address, city, state, zipcode, phone, birthdate
  - juicery: juicery name
  - region: region name
  - users: user name, password
- order entity: represents a purchase of some juice made by a customer
  - order: date, creditcard, expirydate, instructions, customer, juice
  - need credit card with order, rather than customer, because a different card could be used for each order
  - need some way to associate juice with orders

Designing Tables

- customer table
  - customer id
  - name
  - address
- order table
  - order id
  - customer id
  - juice id
  - quantity
- can only order one type of juice in a single order
  - solution: add “juice id2”, “juice id3”, “quantity2”, “quantity3” to the order table
  - must decide on a maximum number of juice per oder
  - must decide on empty values if an order has fewer than this

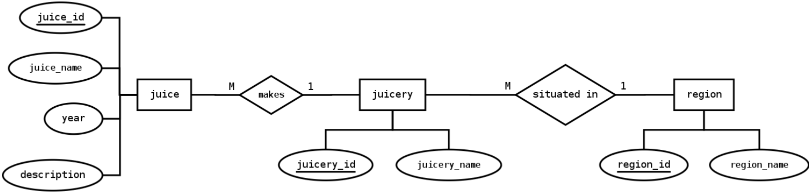
Normalization

- better solution: store the items that make up an order
- items table
    - item id
    - price
    - quantity
    - juice id
    - order id
- when do you add a table versus or more attributes?
    - normalize the database according to a set of rules
    - `MySQL article`

Types of Relationships

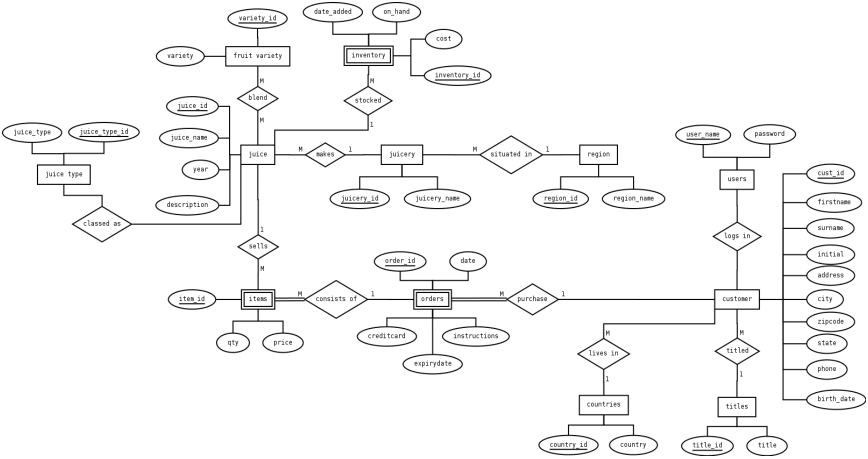
- one-to-one
  - exactly one instance of the first entity for each instance of the second entity
  - example: customer has exactly one set of login information
- one-to-many
  - one or more instances of the second entity for each instance of the first entity
  - example: each juicery sells many juices, but each juice is made by exactly one juicery
- many-to-many
  - each entity is related to more than one instance of the other entity
  - example: a juice can be made up of many types of fruits, and each fruit can be in more than once juice

Examples



- a juicery can make many juices
- each juice is made by one juicery
- similar reasoning for the regions

Complete Example



Primary Keys

- must uniquely identify each record
  - can use a combination of keys, e.g. surname and firstname
  - must be sure it will be unique!
  - can always use an integer identifier
- for orders, use a combination of order_id and customer_id to uniquely identify the order
  - allows each customer to have an order #1
  - this makes orders a weak entity: part of primary key is the key from another entity
  - represented with a double box
- orders have a full participation relationship with customers
  - can't have an order without a customer
  - similar reasoning for items: can't exist without an order
  - represented with a double line