Schedule Database Project

CS342 Fall 2015

Bethany Armitage & Jasjot Sumal

Table of Contents

phase 1 - Information Gathering and Conceptual Database Design

Phase 1 Information Gathering and Conceptual Database Design

Section 1 Fact Finding Techniques and Information Gathering

1.1.1 Introduction to Business

The Sequoia Sandwich Company is a local delicatessen that serves sandwiches, salads, soups, and bakery desserts. They also offer catering for groups in the form of sandwich platters, boxed lunches, along with options to add beverages, chips and side salads. There are currently four locations, three in Bakersfield: downtown, southwest, and Rosedale, plus one in the Fresno/Clovis area.

1.1.2 Description of Fact Finding Techniques

Operating procedures pertaining to the company were discovered primarily through prior experience working for the company in addition to research from their website. Operations on the data will be handled by the person in charge of scheduling which is the branch manager at the location. The database will be used to generate reports describing employees and their current work availability.

1.1.3 Focus of Database

The focus of the database is management of employee work availability and weekly scheduling. The company currently stores its employees' availability on pen and paper. The manager then references this paper to type the weekly schedule in a Microsoft Excel spreadsheet, which uses formulas for counting the current number of days that any given employee has on the schedule being created. This formula is not very comprehensive, as it will not even compensate for misspellings. The manager must manually cross check several scheduling criteria on their own. The employee must availability for the shifts and days that they are scheduled for, they must only be scheduled once per day, they cannot work days

requested off, they may not work well with another employee, and can only work shifts corresponding to roles they know. The schedule created is printed out upon completion, and posted at the work location. This means employees must copy this information down for their own reference, and if the schedule is changed, employees are not aware of the changes and may either show up to a shift they were not scheduled for or not show up for a shift that they need to work.

Managing everyone's shifts is a laborious and error prone task to do by hand. Our database seeks to solve this problem by providing automation of these checks in an employee management system that will track employees and their work availability given the above constraints, so that managers can quickly and accurately schedule employees for the weekly schedule. Employees will also be able to sign into the front end with a different set of permissions, allowing them to view the weekly schedule without have the ability to modify it. This will provide them access to the latest copy of the schedule anywhere they can access the internet.

The main entities needed to represent this employee scheduling database are Employees, Roles (or Employee Positions), and weekly Shifts. Although other entities will be included to represent the remainder of the shop, these are the entities needed to the database's primary purpose. The front end will focus on coordinating employee scheduling with shift openings, in addition to displaying Supply Deliveries to help employees prepare for the day.

Employee entities store basic employee information such as first name, last name, birthday, and phone number. They also store information pertinent to scheduling: dates when employees were employed and unemployed, and the maximum number of hours employees desire per week.

Role entities simply store the different positions an employee can work; the only attribute they store are the names of different positions within the company. Most of the relevant information regarding roles is in their relationship with Employee entities.

Finally, shift entities store information pertaining to a particular time slot for an entire week. If the company needs to have an employee that is a cashier work from 8:00AM to 3:00PM every day of the week, they would define this and it would be stored in the shift entity. So, the company will be able to decide which shifts are necessary to fulfill for the total daily workload. The shift may define different weekend hours, because the workload is different for week days and weekend days. If a shift is only needed during the week days and not on the weekend, the weekend hours will not be defined, and the shift will function as such.

1.1.4 Itemized Descriptions of Entity Sets and Relationship Sets

Entity Sets

The following is a general summary of entities within the database including their names, meaning, attribute names and properties. More detailed descriptions can be found in the next section.

The **employee** entity stores data about current employees.
`employee (id, first_name, last_name,f is_manager, birth_date, max_hours_per_week, phone_number)`

The **position** entity stores data about various positions that employees occupy.
`position (id, title)`

The **shift** entity defines the possible start and end times for weekday and weekend labor.
`shift (id, title, weekday_start, weekday_end, weekend_start, weekend_end)`

The **delivery** entity defines the days expected supplies are to be delivered from various suppliers.
`delivery (id, title, days)`

The **employment history** entity stores employment history of current and previous employees.
`employment_history (date_employed, date_unemployed)`

The **ingredient** entity stores data regarding ingredients used to create food items
`ingredient (id, name, is_available)`

The **menu item** entity stores data regarding food items that may be sold.
`menu_item (id, name, type, price, photo)`

The **transaction** entity stores sales transactions that have been made by the company.
`transaction (id, date)`

Relationship Sets

The following is a general summary of relationships within the database including their names, meaning, attributes, cardinalities, and participation constraints.

`cannot_work_with (employee1_id, employee2_id)`
Relationship of employee to employee
1…N, partial participation

`requests_off (requested_by, responded_by, start_date, end_date, is_approved)`
Relationship of employee to employee
1…1, partial participation

`has_employment_history (employee_id)`
Relationship of employee to employment history
1…N, total participation

`has_availability_for (employee_id, shift_id, days)`
Relationship of employee to employment history
N…N, total participation

`is_scheduled_for (employe_id, shift_id, date)`
Relationship of employee to shift
1…N, partial participation

`has_position (employee_id, position_id, date_acquired, date_removed, is_primary, is_training, max_days_per_week)`
Relationship of employee to position
N…N, total participation

`has_shifts (position_id, shift_id)`
Relationship of position to shift
1…N, total participation

`delivers (delivery_id, ingredient_id, date_delivered)`
Relationship of delivery to ingredients
1…N, partial participation

`used_in (ingredient_id, menu_item_id, quantity)`
Relationship of ingredients to menu item

1…N, total participation

`sold_in (menu_item_id, transaction_id, date)`
Relationship of menu item to transaction
N…1, total participation

1.1.5 User Groups, Data Views, and Operations

The User Groups are primarily defined by the is_manager attribute within the Employee entity. This attribute dictates access to views that allow the user to add information to the database, as opposed to only visualize it. More specifically, User Groups are defined by Employee Roles; in descending order the privilege levels are manager or employee. A manager will have more access to viewing and editing data than an employee.

Managers

- read, create, update and delete **schedules**
- read, create, update and delete **employees**
- read(view requests off), create(submit requests off) and update (approve/deny) **requests off**
- reading, creating, update and delete scheduled weekly **deliveries**

Employees

- read **schedules**
- read (view own requests off only), create (submit requests off), and update (cancel own requests off) **requests off**
- read weekly **deliveries** (shown on schedule)

1.2.1 Entity Set Description

Listed below are the entities with more detailed descriptions about their purpose and attributes. Next to the name of the entity in parenthesis will describe if it is a strong or weak entity. A strong entity will have a unique primary key identifier, while a weak entity will not have it's own primary key. Asterisked attributes are either a primary or foreign key. The specific type of key will be asterisked in the details. In addition to the names of the attribute; their type and range are specified. If a default value exists for the attribute it will be listed. If an attribute is nullable, it will be listed as such, otherwise it is considered not nullable. The attribute will be described as unique if two rows cannot contain the exact same values. If an attribute is not described as unique it is considered to be not unique. The attribute will be described as single or multivalued, and simple or composite.

Employee	**(strong)**
The employee entity describes employee basic information, maximum hours they are willing to work a week, and whether they have manager privileges or not. It also stores system login credentials. Some employees cannot work with others. Employees may request specific days off of work. They have availability for shifts and they are scheduled for shifts. Employees have one or more position, and they have an employment history associated with them.	
*id	int, 0-999, unique, single, simple, *primary key
clock_number	int, 0-999, unique, nullable, single, simple
first_name	varchar(255), single, simple
last_name	varchar(255), single, simple
is_manager	boolean, true or false, default: false, single, simple
birthdate	date, 01/01/1915 - 01/01/2515, default: null, nullable, single, simple
max_hours_per_week	int, 0-40, default: null, nullable, single, simple
phone_number	big int, 1000000000-9999999999, default: null, nullable, single, simple
email	varchar(255), unique, single, simple
password	varchar(255), default: "<first_name> + sequoiapassword" single, simple

Employment History (weak)

There are cases where employees may work for the company for a time, leave, and then return to the company. For example, students sometimes work only summers. Employees who have at least one column with a start date and a null end date are current employees. This also convenient when employees are rehired because their information will be accessible upon reemployment.

*employee_id	int, 0-999, unique, single, simple, *foreign key
date_employed	date, 01/01/1915 - 01/01/2515, default: today, single, simple
date_unemployed	date, 01/01/1915 - 01/01/2515, default: null, nullable, single, simple

Position (strong)

The position entity stores which types of positions employees may assume. The company has different types of workers who have different tasks. Different positions will have different available shifts. For example, and employee may be a cashier, food prep, janitor etc. It is possible for an employee to be trained in more than one position.

*id	int, 0-999, unique, single, simple, *primary key
title	varchar(255), unique, single, simple

Shift (strong)

The shift entity describes recurring start and end times for when employees are supposed to show up for work. There may be different hours for week days and weekend days. For example, if the company needs one cashier to work everyday in the morning, they would create a shift so that a cashier may be scheduled every day in the morning. They may choose for the cashier to work at 8:00AM and leave at 3:00PM on weekdays and 9:00AM to 4:00PM on weekend days. An employee may have availability for shifts and may be scheduled for shifts.

*id	int, 0-999, unique, single, simple, *primary key
title	varchar(255), default: “shift”, single, simple
weekday_start	timestamp with timezone, 00:00-24:59, nullable, single, simple
weekday_end	timestamp with timezone, 00:00-24:59, nullable, single, simple
weekend_start	timestamp with timezone, 00:00-24:59, nullable, single, simple
weekend_end	timestamp with timezone, 00:00-24:59, nullable, single, simple

Delivery	(strong)
The Delivery entity describes current recurring supply deliveries expected on regular days. The company has several supply shipments a week from several other companies. The delivery will affect the expected daily workload for employees, so it is convenient to know when expected deliveries occur.	
*id	int, 0-999, unique, single, simple, *primary key
title	varchar(255), unique, single, simple
days	int, 0-127, default: 0, single, simple

Ingredient	(strong)
Ingredients are what are used to prepare food items. The company uses several different ingredients in different combinations. Sometimes ingredients are not available. Examples would be lettuce, tomato, various meats, various cheeses etc.	
*id	int, 0-999, unique, single, simple, *primary key
name	varchar(255), unique, single, simple
is_available	boolean, true or false, default: false, single, simple

Transaction	(strong)
A transaction consists of menu items. It is when a customer purchases items.	
*id	int, 0-99999, unique, single, simple, *primary key
date	date, 01/01/1915 - 01/01/2515, default: null, nullable, single, simple

Menu Item	(strong)
Menu items are made of several ingredients and one or more menu items are sold in a transaction.	
*id	int, 0-999, unique, single, simple, *primary key
name	varchar(255), unique, single, simple
type	varchar(255), single, simple
price	number(* ,2), 0.00-99.99, single, simple
photo	lob, 0 gigabytes -2giga bytes, default: 0, single, simple

1.2.2 Relationship Set Description

1.2.3 Related Entity Set

(no related entity sets exist)

1.2.4 E-R Diagram