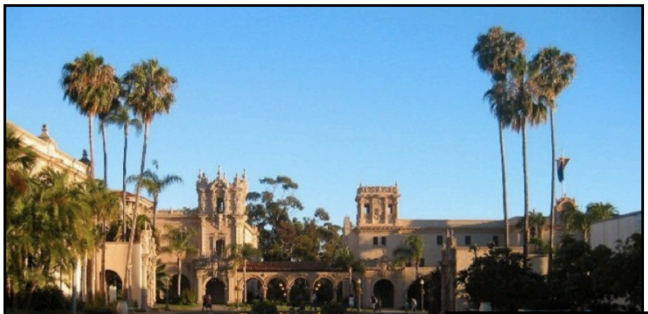
Group Project
E-Tourism Database

Online Booking of Tourist Attractions
User and Employee Management



CS 425 Fall 2012
Database
Organization

Group Members:

Sayali Chandak
(A20287456)

Tegal Gajare
(A20287489)

Matt Jankowiak
(A20279944)

Design Documentation

Table of Contents:

1. Introduction

This project demonstrates the process of building and maintaining a complex database that can be used by a company to advertise certain Tourist Attractions around the US, and offer tickets to each attraction. The database can support any user interface accessible to the general public. For this project we created simple website.

2. Group Roles

The following shows a breakdown of the roles in our group. While we did work on these aspects of the project separately, we often corresponded and worked collaboratively on each.

Sayali Chandak (A20287456):
- Conversion of SQL queries to java
- Creation of database tables
- General Processes, SQL Queries

Tejal Gajare (A20287489):
- Creation of front end website
- Integration of java into jsp
- General Processes, SQL Queries

Matt Jankowiak (A20279944):
- Populate all tables in the database with workable data
- General Processes, SQL Queries
- Design Documentation and Test Data

3. Organization of the Database

The database provides the ability to maintain three major aspects of the business:

1. To keep track of the various attractions offered on the website. Each attraction has the following information available to the user:
- City, location, address, contact info, website
- Brief description and type of attraction
- Discussions and reviews written by users
- Ability to purchase tickets to an attraction
- Suggested itinerary if a user wants to visit multiple attractions in one day

2. To organize users of the website. There are two types of users: Registered and Guest. Registered users have the ability to do the following:
- View attractions in a particular city and purchase tickets to attractions
- Earn reward points by purchasing tickets
- Gain different membership statuses with reward points
- Use reward points earned instead of cash to purchase tickets to other attractions

Guest users are able to do the following:
- View all attractions and discussion forums, but they are not allowed to purchase tickets or add to discussions without registering through the website.
- Become a registered user by providing details such as username, password, etc.

3. To organize employees of the company. This is a complex aspect of the database that includes the following:
- A list of all employees, their personal information, and their role within the company
- A schedule of employees' workdays and the days they are on leave
- The minimum staff required for each attraction
- Designation of different employees to different projects.

4. Explanation of Processes on the Website

The website provides access for both Users and Employees. On the main page of the website, a visitor can enter a username and password and then select whether he wants to log in as a User or Employee. Additionally, a visitor can log in to the website as a Guest without having to enter login credentials.

If the visitor signs in as a Registered User, he is shown basic information about his user such as his name, membership status, and reward points. The Registered User can then view attractions, purchase tickets, begin or add to discussions, and change his phone number.

When an employee logs in to the website, he has different options depending on the type of employee. For instance, if the CEO logs in, he can change the staffing level of each attraction. When a Director logs in, he will be shown which employees are working under him, their schedules of the past two days, and if there are no security officers scheduled to work tomorrow.

If a visitor chooses to log in as a Guest, he still has the ability to view Attractions and discussions on those attractions. The Guest can then easily sign up as a Registered User if he would like to purchase tickets or comment on Discussions.

5. Table Structure and Schemas:

The following is the order of dependencies of the tables in our database. Tables 1 – 12 are not dependent on other tables. Tables 13 – 31 have Foreign Keys that depend on tables with lower numbers. Any given table might be dependent on other tables that have a lower number, and there are no tables dependent on others that have a higher number. These tables have been updated based on HW3 feedback.

1. Registered_User (username, password, first_name, last_name, user_id, phone, address, credit_card_num, email_id, dob, membership_status, current_reward_points, recent_activity)

2. Guest_User (guest_id, username)

3. User_Info (user_id, user_type)

4. Points (username, earned, status)

5. System (user_role)
Attribute user_role may be "User" or "Employee"

6. Company (company_id, name)

7. Employee (id, ssn, name, addr, phone, email, city, role, abilities_multi, username, password)

8. Emp_Calendar(emp_id, start_date, end_date)

9. Emp_Leave (emp_id, start_date, end_date)

10. Mapping_employee_type(emp_type_no, emp_type)

11. Complete_Purchase_History (ticket_number, price, user_id)

12. Tickets (ticket_number, price, quantity, points_rewarded, valid_date, start_time, end_time, ticket_id)

13. Calculate_Sale (id, ownership_id, commission, sale)
FK : ownership_id references company_id in Company

14. Project (id, project_name, emp_id, start_date, end_date)
FK : emp_id references id in Employee

15. CEO (emp_id)
FK : emp_id references id in Employee

16. Marketing_VP (emp_id, project_id)
FK : emp_id references id in Employee
FK : project_id references id in Project

17. Chief_Editors (emp_id, project_id)
FK : emp_id references id in Employee
FK : project_id references id in Project

18. Associate_Editors (emp_id, project_id)
FK : emp_id references id in Employee
FK : project_id references id in Project

19. Team_Leader (emp_id, function, project_id)
FK : emp_id references id in Employee
FK : project_id references id in Project

20. Operational_VP (emp_id, project_id, supervisor)
FK : emp_id references id in Employee
FK : project_id references id in Project

21. Directors (emp_id, location, supervisor)
FK : emp_id references id in Employee

22. Ticket_Sellers (emp_id, project_id, supervisor)
FK : emp_id references id in Employee
FK : project_id references id in Project

23. Security_Guard (emp_id, project_id, supervisor)
FK : emp_id references id in Employee
FK : project_id references id in Project

24. Tour_Guides (emp_id, project_id, supervisor)
FK : emp_id references id in Employee
FK : project_id references id in Project

25. Maintenance_Staff (emp_id, project_id, supervisor)
FK : emp_id references id in Employee
FK : project_id references id in Project

26. Itinerary (itinerary_id, start_time, end_time, registered_user, city, attractions_multi, att_first, att_second, att_third)
FK: registered_user references username in Registered_User

27. Attraction (id, name, city, address, phone,start_hours, end_hours, email, best_times, ownership, director, ticket, limit, itinerary, start_time, end_time, description, instructions, min_staff)
FK: ticket references ticket_number in Tickets
FK: itinerary references itinerary_id in Itinerary
FK: ownership references company_id in Company
FK: director references emp_id in Director
instructions nullable
Attribute best_times may be “Spring” or “Summer” or “Fall” or “Winter”

28. Ticket_Package (package_id, package_price, dayseller_offered, city, registered_user, itinerary, attractions_multi)
attractions_multi multivalued
FK: registered_user references username in Registered_User
FK: attractions_multi references id in Attraction

29. Review (id, review_title, rating, limited_comment, attraction_id, date, user, review_like_cnt, review_dislike_cnt, attraction_like_cnt, attraction_dislike_cnt)
FK: attraction_id references id in Attraction
FK: user references username in Registered_User

30. Discussion_Forum (comment_id, thread_id, preset_topic, thread_title, comment_description, attraction_id, user_id, city, port_date, like_cnt, dislike_cnt, sr_no)
FK: attraction_id references id in Attraction
FK: user references user_id in User_Info

31. Staffing_needed (emp_id, project_id, supervisor)
FK: attraction_id references id in Attraction
FK: project_id references id in Projects
FK: emp_id references id in Employee

6. 3NF Proof

The following proofs show that the tables in our database meet 3NF requirements and that the tables are lossless. The order of the following proofs is in the same order as the schemas shown above. Each foreign key reference contains a proof.

Company ∩ Calculate_Sale == ownership_id → Primary key in Company

Employee ∩ Project == id → Primary key in Employee

Employee ∩ CEO == id → Primary key in Employee

Employee ∩ Marketing_VP == id → Primary key in Employee

Project ∩ Marketing_VP == id → Primary key in Project

Employee ∩ Chief_Editors == id → Primary key in Employee

Project ∩ Chief_Editors == id → Primary key in Project

Employee ∩ Associate_Editors == id → Primary key in Employee

Project ∩ Associate _Editors == id → Primary key in Project

Employee ∩ Team_Leader == id → Primary key in Employee

Project ∩ Team_Leader == id → Primary key in Project

Employee ∩ Operational_VP == id → Primary key in Employee

Project ∩ Operational_VP == id → Primary key in Project

Employee ∩ Directors == id → Primary key in Employee

Employee ∩ Ticket_Sellers == id → Primary key in Employee

Project ∩ Ticket_Sellers == id → Primary key in Project

Employee ∩ Security_Guard == id → Primary key in Employee

Project ∩ Security_Guard == id → Primary key in Project

Employee ∩ Tour_Guides == id ➝ Primary key in Employee

Project ∩ Tour_Guides == id ➝ Primary key in Project

Employee ∩ Maintenance_Staff == id ➝ Primary key in Employee

Project ∩ Maintenance_Staff == id ➝ Primary key in Project

Registered_User ∩ Itinerary == username ➝ Primary key in Registered_User

Tickets ∩ Attraction == ticket_number ➝ Primary key in Tickets

Itinerary ∩ Attraction == itinerary_id ➝ Primary key in Itinerary

Company ∩ Attraction == company_id ➝ Primary key in Company

Director ∩ Attraction == emp_id ➝ Primary key in Director

Registered_User ∩ Ticket_Package == username ➝ Primary key in Registered_User

Attraction ∩ Ticket_Package == id ➝ Primary key in Attraction

Attraction ∩ Review == id ➝ Primary key in Attraction

Registered_User ∩ Review == username ➝ Primary key in Registered_User

Attraction ∩ Discussion_Forum == id ➝ Primary key in Attraction

User_Info ∩ Discussion_Forum == user_id ➝ Primary key in User_Info

Attraction ∩ Staffing_needed == id ➝ Primary key in Attraction

Projects ∩ Staffing_needed == id ➝ Primary key in Projects

Employee ∩ Staffing_needed == id ➝ Primary key in Employee

7. Changes made based on HW3 feedback:

Based on feedback given after HW3, we have added 3NF proofs in this design document. Our last set of schemas in HW3 did meet 3NF requirements, however we were missing the proofs showing such. 3NF proofs are in section 6 of this document.

We also received feedback suggesting that we add a “Cities” table. Based on the structure of our database, we do not need a table for Cities, as city information for each Attraction is given as a parameter in the Attraction table. We are still able to meet every requirement of the Project Deliverables in regards to cities without a City table. A user is able to select a city from a drop down menu on the website to see which Attractions are in that city.

We have changed out schemas related to discussion forums. In HW3, we did not have the ability to have a parent thread with multiple child threads. In our Discussion Forums table, we have the attributes Comment_Id,Thread_ID, and SR_NO to handle this.

We have added a two tables to handle employee’s working days: Emp_Calendar and Emp_Leave. The first shows a range of dates that a particular employee is working. The second shows specific days an employee is on leave. With these tables we are able to handle queries 6 and 7.