***Group #3***

***Members: Asma - Zackline - Nina***

***Project Definition***

***Due date: 11/1/2016***

***Course#: CSC 621***

***Description of the Project***

*Company information:*

Dayesda (Emergency Management Company) is the company that we will do our project about. Dayesda is a startup company that provides enhanced emergency management *services* and *courses* in Saudi Arabia. Its services and courses are goal-oriented, cost-effective, and efficient. It has expertise and affiliation with U.S emergency companies and expertise from around the world.

*Examples of Service Beneficiaries:*

* Schools, hospitals, industries.
* Civil defense.
* Saudi Red Crescent Authority.
* Saudi Humanitarian Emergency Aid and Response Team (Saudi HEART).
* Ministries.
* Any Private sector who deals with hazards materials.

*Examples of CourseBeneficiaries:*

* Civil defense workers.
* Saudi Red Crescent Authority workers.
* Saudi Humanitarian Emergency Aid and Response Team (Saudi HEART) workers.
* EMTs and Paramedics.
* Professionals and healthcare providers who are working in emergency management positions.
* Officials who working the Internal Ministry in positions relating to security and emergency management.
* Individuals who work in Ministry of Hajj as emergency planners and coordinators.
* Ministry of Health managers who are working in the Emergency Management section.
* Workers who deal with hazards materials.

*Services:*

1. Emergency management.
2. EOP(emergency operation plan)
3. Business continuity plans.
4. COOP(continuous of operation plan)
5. Infrastructure protection planning
6. Disaster response planning
7. Hazard mitigation planning
8. Hazard recovery planning
9. Preparedness planning
10. Hazard Vulnerabilities Assessment planning
11. Threat and Hazard Risk Assessment planning
12. Exercises and drills management planning
13. Walk in security assessment
14. Incident Response
15. Exercises and drills management evaluation
16. Infrastructure protection evaluation.
17. Disaster response evaluation
18. Hazard mitigation evaluation
19. Hazard recovery evaluation
20. Damage Assessment.

*Courses:*

1. CERT(community emergency response team)
2. Disaster Life Support (DLS)
3. Public Information Officer (PIO)
4. Disaster Rapid Initial Assessment
5. Counter-terrorism procedures.
6. ICS 300(incident command system)
7. ICS 400(incident command system)
8. Planning for children in disaster
9. FEMA course (10 courses from Fema 1 to Fema 12)

*DB requirements:*

The DB that we want to create for this company will cover the online service requesting process and the online course registration. Therefore, our data base will keep tracks of the company Employees, the services that the company provides for organizations, and the courses that the company provides for individuals, also the organizations’ information and the trainees’ information who benefit from this company Also, it will keep tracks of the course payments that are made by the trainees. We don’t need to track the organization payments since it done after agreement and paper work, we will just store the price of requesting a service by an organization. So, the users of our data base are (Organizations, Trainees and Employees of our company).

* For each Employee, DB keeps track of unique employee identifier, name, address, and phone, his role (job) in the company, salary, and password.
* For each service, DB keeps track of unique service identifier, service name, service description (there are no more important characteristics other than these since the whole service depends on employee checking and paper work.
* For each request of a service, DB keeps track of unique request identifier, date of request and cost of conducting this specific service for this particular organization. The employee will visit the organization to check and do the work (service) according to that.
* For each organization requests a service, DB keeps track of unique organization identifier, organization name, and sector (private or public), location, phone, password.
* For each Trainee takes a course, DB keeps track of unique trainee identifier, name, and phone, password.
* For each course, DB keeps track of unique course identifier, course name, description, and course tuition, duration (number of weeks).
* For each payment by the trainee for a course, DB keeps track of unique card number, card type, and the billing address.

*Type of Queries Users will run:*

1. Insert queries:

All our different users will be able to insert into the different relations

**Employees:** Will be able to insert a record into the employee relation when they register.

**Organization:** Will be able to insert a record into the organization relation when they register.

**Trainee:** Will be able to insert a record into the trainee relation when they register.

1. Update queries:

Our users will be able to make update in their various relations.

Let’s say an Employee, Trainee, or Organization has a name, phone number or address change, they should be able to make that update which will be captured in the relations. But a trainee will not be able to make any update in the Organization or employee relation. An employee can make update in the Organization relation, that is, make an update of the price for the service rendered to a particular organization.

1. Select Queries:

Our users will have the ability to access only the data that is related to them and nothing else.

For example a trainee will be able to access information about himself and the courses he or she is enrolled in. An employee can also access information about both the trainee and the organization especially when they need to make update in the price attribute of the request record.

We will not give our users the ability to create relations, drop relations or even delete records from our relation because we are particular about our data integrity and giving our users too much access will cause us lots of issues.

*Types of queries DB supports:*

1. Create queries :
   * Create queries for all DB relations(employee, organization, trainee, course, service, request, payment, is assigned to, completion)
2. Insert queries :

* When an organization requests a service it will be inserted into the database as a request record.
* When a trainee enrolls in a course it will be inserted into the database as a completion record. Also it will insert its payment transaction in payment relation.
* Insert into trainee and organization relations.
* Insert all employees of the company in the employee relation
* Insert all services into service relation
* Insert all courses in course relation.

1. Update queries:

* For each service request the employee need to update the price in the request record according to the agreement with the organization after visiting.
* Any update on any record of relations (employee, organization, trainee, course, service).

1. Delete queries:

* Delete any request transaction if the company didn’t agree with the organization about the price or for any problem.

1. Read Queries(report):

* Each trainee can check on courses information that he had enrolled (e.g. tuition, instructor).
* Each employee can check on service request information that he is assigned to.
* Find the organization that request the most services for any privilege in the future.

The delete and create queries are accepted only for us the developers.

*Referential integrity constraints:*

* Each unique request record contains two FKs (both Not Null), one matches one of the values in the PK column of the organization relation, and the other matches one of the values in the PK column of the service relation.
* Each record in a relation (is assigned to), that represents the relationship between request of the service and employee, contains two FKs (both Not Null) one matches one of the values in the PK column of the employee relation, and the other matches one of the values in the PK column of the request relation.
* Each unique course record contains FK (Not Null) matches one of the values in the PK column of the employee relation, which is the employee id who teaches this specific course.
* Since each trainee can takes the same course multiple times in different years. Each record in completion relation contains partial Identifier (year of completion) and two FKs (both Not Null) one matches one of the values in the PK column of the course relation, and the other matches one of the values in the PK column of the trainee relation. All together form the PK of this relation.
* Also, each unique completion record contains FK (Not Null) matches one of the values in the PK column of the Payment relation.
* All FKs in our relations are restricted to delete and cascade to update, but we will modify that if we face any problem while working.

*How data will be accessed:*

* Data in all of our relations will be accessed using the sql select statements. Of course we will not have our users write and run sql queries, instead in development, we will create the back end which is the database and write queries to run off the database.
* We will have front end application where the users can enter the required credentials to access the data.
* When the user enters the values, then the service which we using php, will link the front to the back end and search in the back end, the relations where it can find a value or values that matches the requirement the user enters.
* If a record or records are found that matches the criteria then the record or records are returned and displayed to the user else and empty set or error is thrown out.
* For each form or forms included in the application we may have to select from either one of our 9 relations or a join of multiple of the relations depending on the data that needs to be accessed.

*Indexing:*

* It is important that we put a mechanism for increasing the speed of our data search and data retrieval in place so as not to allow our users having to wait so long just to access and retrieve the information they need and that is where indexing comes in handy.
* On the other hand indexes also cost a lot, requires too much of time and resources to maintain, therefore it is wise only to use them when you have large volume of data to access, and if adding them speeds the data access and retrieval process.
* Dayesda, is a startup company and our data is not yet in large volume to the point of accessing and retrieving it takes a long time. Besides, the time to access the data without an index is approximately equal to the time to access and retrieve the data using an index. And since we are particular now about the cost of the index, we are not going to use them. We will consider using them in the future only if they are our only option and we need to use them, in that case, we will use mostly (Organization name, Employee name, and Trainee name) to be our future indexes since their selectivity is acceptable.
  + *ALTER TABLE organization ADD INDEX (orgname);*
  + *ALTER TABLE employee ADD INDEX (fname);*
  + *ALTER TABLE trainee ADD INDEX (tfname);*

*Work Schedule (whole team):*

|  |  |
| --- | --- |
| Date | Work |
| Week 1: Oct 20-Oct 31 | Project definition- choosing the idea and agree on it. Working on collecting/defining the requirements of our DB/ discuss the vision of our DB to imagine the design/ set up GitHub repository for project. |
| Week 2: Nov 1-Nov 7 | ERD/relational Model/ Indexing/Functional dependencies. |
| Week 3: Nov 8-Nov 14 | SQL of creating and population the DB. |
| Week 4: Nov 15- Nov 21 | Working on queries and their interfaces. |
| Week 5: Nov 22- Nov 28 | Error checking/ Presentation. |
| Week 6: Nov 29- Dec 5 | Finalize the final paper/project/presentation. |
| Week 7: Dec 6 | Final Project paper/ final presentation. |

*Work Schedule (Asma):*

|  |  |
| --- | --- |
| Date | Work |
| Week 1: Oct 20-Oct 31 | * Write the description of the database requirements and the different users of DB and their requirements after discussion with team. |
| Week 2: Nov 1-Nov 7 | After discussion with team:   * Write the Description of the referential integrity constraints. * Creating the ERD. * Creating the Relational Model. |
| Week 3: Nov 8-Nov 14 | * SQL for creating tables of the DB. * SQL for population 4 tables. |
| Week 4: Nov 15- Nov 21 | * SQL of two queries and their interfaces (with screenshots). |
| Week 5: Nov 22- Nov 28 | * Doing 50 % of presentation. |
| Week 6: Nov 29- Dec 5 | * Finalize |

*Work Schedule (Zackline):*

|  |  |
| --- | --- |
| Date | Work |
| Week 1: Oct 20-Oct 31 | * Write the list of tasks for the project and discuss the definition of the project with team and. |
| Week 2: Nov 1-Nov 7 | After discussion with team:   * Write on index selection. * Write about the FD and how it is normalized in the project. |
| Week 3: Nov 8-Nov 14 | * Revise SQL for creating tables of the DB. * SQL for population 3 tables. |
| Week 4: Nov 15- Nov 21 | * SQL of two queries and their interfaces (with screenshots). |
| Week 5: Nov 22- Nov 28 | * Doing 50 % of presentation. * Work on ensuring integrity after discussion that with team. |
| Week 6: Nov 29- Dec 5 | * Finalize. |

*Work Schedule (Nina):*

|  |  |
| --- | --- |
| Date | Work |
| Week 1: Oct 20-Oct 31 | * Discuss the definition of the project with team and revise the project description. |
| Week 2: Nov 1-Nov 7 | After discussion with team:   * Work on FD closure test. |
| Week 3: Nov 8-Nov 14 | * Revise ERD and Relational Model for any update. * SQL for population 2 tables. |
| Week 4: Nov 15- Nov 21 | * SQL of two queries and their interfaces (with screenshots). |
| Week 5: Nov 22- Nov 28 | * Revise presentation. * Revise final Project paper. |
| Week 6: Nov 29- Dec 5 | * Finalize. |