University of British Columbia, Department of Computer Science

CPSC 304

Cover Page for Project Part 1

Date: September 26, 2018

Group Members:

Name	Student Number	CS Userid	Tutorial Section	Email Address
Damasia Maria Correch	28649151	o2e1b	T1C	damacorrech@gmail.com
Naitong Chen	21539151	q7l0b	T1G	chennaitongubc@hotmail.com
Chloe You	11654150	i0e1b	T1D	chloeyxy@outlook.com
Christina Cheung	37405157	d8f1b	T1D	ccheung256@gmail.com

Project Formal Specification

Change in ER Diagram & Relational Schema:

1. add phone number and email address as attributes to LabMember on ER diagram/Relational Schema
2. change *Collaborator* table: "ON DELETE SET NULL"
3. change *Open Grant* table: "ON DELETE CASCADE"
4. add "status" attribute to "applies" relationship (rejected, pending, approved)to the *Applies_OpenGrant* Table
5. Change "*Applied Grant*" entity name to "*Approved Grant*"

Platform: CS UGrad Oracle installation and JDBC

Users:

- DB admin
    - Sees everything in the database including Grants
    - Insert LabMembers
    - Insert Open Grants
    - Delete Open Grants where Applies Open Grants status = "approved" (when status of any Applies Open Grants relationship with an Open Grant is changed to "approved", DB admin should move said Open Grant to Approved Grant by deleting the Open Grant and inserting an Approved Grant with the same information)
    - Insert Approved Grants
    - Delete entries in Applies Open Grants
    - Delete Collaborators
    - Update LabMember name, ID, and education
- LabManager
    - Sees everything in the database including Grants
    - Insert Subjects
    - Update Subject name
    - Update Subject availability
    - Update Subject id
    - Insert Collaborators
    - Insert Booking
    - Update Booking
    - Update Applies Open Grant status (applied, approved, rejected)

- PI
    - Sees everything in the database including Grants
    - Update Project name
    - Update Project category
    - Update Project material type
    - Insert Collaborators
    - Distributes funding (this means the PI can add entries to the Approved Grant Fund table)
- Researcher
    - Sees everything in the database except Grants
    - Update Project name
- Collaborator
    - Sees only Projects that they are assigned to

Queries:

- Query list of *Subjects* in each *Project*
    - project *Subject* **sid** given *Project* **projectName** (selected from *Participates* table)
- Query list of available *Subjects* with requested **startTime** and *Booking* **length**
    - project **sid** of all *Subjects* selecting availability = true AND whose existing *Bookings* are not in conflict with queried booking. "In conflict" means that the requested startTime and length would overlap with an existing *Booking* startTime and length. (availability is whether *Subject* is willing to be booked, startTime and length are both Ints)
    - For this we must **JOIN** *Subject* table (to get **sid** and **availability**) with *Participates* table (to get **startTime**) and *Takes_Booking* table (to get **length**)
    - Exclusive to LabManager
- Query Researchers in each *Project*
    - project *Researcher* **rid** given *Project* **projectName** (selected in *Supervises_WorksOn*)
- Query collaborators in each *Project*
    - project *Collaborators* **id** given *Project* **projectName** (selected in *Assigned_Collaborators*)
- Query which grants each project has
    - project *Grant* name given *Project* **projectName** (selected in *Fund_ApprovedGrant*)
- Count total # of projects
    - count all entries of *Project*
- Get total $ grant received for project
    - select **projectName** in *Fund_ApprovedGrant*, project **amount** and sum them all up

- Get how much funding is left in a project after materials are purchased (assuming we only purchase once, at the start of the project)
    - For this we must **JOIN** *Fund_ApprovedGrant* (to get **amount**) and *Project_MaterialType* (to get **name**) and *MaterialType_MaterialPrice* (to get **materialPrice**).
    - The result of the join is subtracted from the total $ grant received (query described above)
- Identify whether a participant number was in the control or experimental condition
    - project **participantTestCondition** given **participantNumber** and **projectName** (selected from *Takes_Booking*)
- Get number of participants in each condition for a project
    - group *Bookings* by the **participantTestCondition** (**GROUP BY COUNT**, selecting for given **projectName**)
- Query the expertise of researchers in each project
    - project **expertise** and **name** given **projectName** (selected from *Supervises_WorksOn*)
    - For this we must **JOIN** *Researcher* (to get **expertise** and **name**) and *Supervises_WorksOn* (to get **projectName**)
- Query the number of researchers with a specific expertise given a project
    - group researchers by their expertise (**GROUP BY COUNT**) using the query above (joining *Supervises_WorksOn* and *Researcher*)
- Query total number of weekly hours allocated by a given researcher
    - project **weeklyHoursAllocated** for given *Researcher* **rid** (selecting from *Supervises_WorksOn*), then sum all hour values.

View:

- Researcher and LabManager's view of LabMembers is restricted: they can see all attributes for LabMembers of their own Lab (same Lab ID), but can only see name and education for LabMembers in other Labs
- Collaborator's view on *Project* is restricted to the project they are assigned to, they can see all attributes of *Project*
- Collaborator's view on *Supervises_WorksOn* is restricted to the entries whose project is one that they are assigned to (given **projectName**)

Division of Labour:

- Create tables and populate with data (Christina)
- Code queries and test in SQL (split evenly)
- Each person tests their own queries and then all of the queries
- Application implementation (Chloe, Naitong and Dama)
- Embedding the SQL statements in the application (split evenly)
- Documentation (split evenly)