DesignProcess-QuarryHill(DAP&RM).docx

Final Project – Fall 2019

CS385 – Applied Database Management Systems

Doug Plager and Riley Morrow

Due date: 12/10/2019, 9 am

**Design Process for Quarry Hill registration database:**

**I. Waterfall / T-A-R-C + E-R Diagram => Table schemas and Normalization approach:**

[ where “T-A-R-C” refers to Tables(Entities)-Attributes-Relationships-Constraints(PK, FK, and Cardinality Ratios) ]

1. Specify Requirements and Outline (from Assignment sheet and Quarry Hill Brochure and forms)

Tables (Entities) – Noun-like “things”

Attributes – associate with a Table/Entity or just list

Relationships – Verb-like connections b/w Entities

1. Generate E-R Diagram Framework (Entities and Relationships w/o attributes)

NOTE: We thought about which entity(s) and relationship(s) are “central” to the database. They will likely need to have a single(or minimal-composite) primary key attribute, and they will likely correspond to table(s) with the most rows. In this case, our CampSessions – Registration – Campers (entity-relationship-entity) group appeared to be central to our E-R Diagram.

1. Consider Attributes and their Table/Entity locations (add attributes to preliminary Table Schemas)

Consider potential Primary Keys (PKs).

1. Consider E-R Connections and Mapping Cardinality Ratios (add to E-R Diagram)

Consider potential Foreign Keys (FKs).

1. Generate Preliminary Table schemas while thinking about Mapping of E-R Diagram into Tables, Steps A – F (below).
2. Consider whether the current model will address likely queries.
3. Normalize preliminary Tables (1NF, 2NF, 3NF), including functional dependency considerations.

**II. Mapping E-R Diagram into Tables:**

[CS385 11/12/2019 Class Notes (and textbook Chap. 7 & 7.7.4)]

Steps A through F:

(A) Regular Entities:

(i) create a corresponding Table with atomicity (break down composite attributes) and singularity (multivalued attributes) in mind.

(ii) select a Primary Key.

[ thus, approaching 1NF ]

(B) Multivalued Attributes: [ NOTE: We addressed multivalued attributes early. ]

(i) create a separate Table for the multivalued attribute and any associated, functionally-dependent attributes.

(ii) include as a Foreign Key attribute in this new Table, the Primary Key attribute (using the same or a different column name) of the original Table.

(C) 1-to-1 Relationships:

[ NOTE: Consider the Instructors(entity) – Advise(relationship) – Students(entity) example with respect to mapping 1-to-1, 1-to-Many, and Many-to-Many relationships ]

(i) a 1-to-1 entityA-relationship-entityB arrangement maps to TableA and TableB with each of the associated attributes (including “descriptive attributes” of the relationship) going within the most appropriate of the two tables.

(ii) put a Foreign Key attribute in one of the tables referencing back to the other table, with the FK being placed in the table with total participation under such a circumstance.

(D) 1-to-Many Relationships:

(i) a 1-to-Many entityA-relationship-entityB arrangement maps to TableA and TableB, with any “descriptive” relationship attributes added to the Many-side table (TableB).

(ii) put a Foreign Key attribute in the Many-side table (TableB) referencing back to the other, 1-side table (TableA).

(E) Weak Entities:

[ NOTE: Consider the course(entity) – sec\_course(relationship) – section example in our Database System Concepts, 6th Ed. textbook, Figure 7.15 (E-R Diagram) and Figure 2.8 (Relational/Table Schema diagram) ]

(i) it appears that the PK attribute of the strong entity (‘campID’ of ‘Camps’) that is associated with the weak entity (‘CampSessions’) should be included as a FK attribute in the table (also ‘CampSessions’) mapped from the weak entity.

(ii) identify a PK in the table derived from the weak entity (i.e., CampSessions); this compositie weak-table PK will include the strong entity’s PK, or as in our Quarry Hill project, we “combined” the composite PK into a single-attribute PK (which seemed to work).

(F) Many-to-Many Relationships:

(i) a Many-to-Many entityA-relationship-entityB arrangement maps to TableA, TableR, and TableB with each of the associated attributes (including “descriptive attributes” of the relationship) going within the most appropriate of the three tables.

(ii) the table derived from the relationship (TableR) will include the PK attribute from each of the two other tables (i.e., from TableA and TableB) to form a composite PK for TableR.

(iii) each of the two PK attributes from TableA and TableB should be designated as an FK in TableR referencing back to its respective table (either TableA or TableB).

**Quarry Hill(QH) database Challenges and Decisions:**

Challenge 1: Whether to have a separate ‘Fee’ Entity and ‘Payment’ Relationship, which we initially included in our 1st E-R Diagram?

Decision: Eventually decided to put the Fees information as attributes within our ‘Camps’ table and the Payment information as attributes of the ‘Registration’ table (which arose from our Many-to-Many ‘Register’ relationship). If each significant change to a camp is captured as a new ‘Camps’ tuple, this should allow us to better track the cost of each camp over time and avoid the potential for a “historical” modification anomaly if, e.g., a camp’s ‘memberFee’ and ‘nonmemberFee’ values are changed in the future relative to a past unpaid camp fee.

Challenge 2: How best to handle camp session date changes from year-to-year?

Decision: It seemed like we should be able to limit to 7 different potential week-long date ranges for any given camp, but conversely, it seemed as though many other potential changes to the precise date range (‘dateRange’) for a given camp session could occur over time (e.g., a camp running only 3 days instead of 5 days). Furthermore, the ‘year’ for each camp session would need to change each year to maintain good historical data integrity. Thus, as for Challenge 1, we decided to explicitly indicate the date range and year for each of the CampSessions tuples. The downside to this appears to be the need to generate a new set of camp sessions each year.

Challenge 3: How to handle the ‘CampSessions’ weak entity?

Decision: We essentially handled a 1-to-Many Camps(entity) – Subdivide(relationship) – CampSession(weak entity) arrangement as a “1-to-Many Relationship”. In addition, we combined our weak entity-derived CampSessions’ three-attribute, composite PK (campID, sessionID, year) into a single-attribute PK (campSessID)), which helped simplify queries involving the ‘CampSessions’ table.

Challenge 4: How to properly handle a session change and/or a session cancellation?

Decision: Along with camp fee payment-related attributes (see Challenge 1, above), we decided that session change- and session cancellation-related attributes would fit well in our ‘Registration’ table. By doing this, we thought that subsequent queries to determine proper “completion” (i.e., full payment, change, or cancellation) of each registration event would be more straightforward, even though a registration change would require generation of a new Registration tuple (a new tuple that still indicates that the $2.50 registration fee has been paid, but payment of the $10 change fee would be indicated in the original registration event).

Challenge 5: Single vs. Multivalued attributes?

Decision:

1. we decided that storage of a single contact email and phone number for each adult contact and instructor would be sufficient in this case.
2. as suggested in Challenge 4, we decided that a single registration change could be associated with each registration event while maintaining the ability to query for all registration events for a single individual to track the final outcome.
3. we decided that specific requirements for a given camp needed to be treated as a multivalued attribute because a few of the camps definitely had multiple specific requirements; thus, we added an “intervening” ‘CampsReqs’ table (arising from our E-R diagram’s Many-to-Many ‘Require’ relationship) and a separate ‘Requirements’ table (NOTE: In hind sight, we probably should have grouped all “general requirements” (protection against wet, dirt, sun, and insects and wearing of athletic shoes) into a single ‘reqdItem’ within the ‘Requirements’ table].
4. similar to (c) just above, we decided it would be important to handle the potential for a camper’s multivalued medical information, so we added a ‘MedConditions’ table (arising from our E-R diagram’s 1-to-Many ‘Treat’ relationship).

**Compromises with respect to 3NF:**

NOTE: It appears that all of our tables meet 1NF and 2NF.

(1) For the 3-4 y/o single-day camps of our ‘Camps’ table there is transitive dependency of ‘memberFee’ and ‘nonmemberFee’ on ‘camperAgeGrade’; however, this transitive dependency breaks down for the camps for other ages or grades and one might anticipate that differences in the memberFee and nonmemberFee values among the 3-4 y/o single-day camps could arise in the future.

(2) In our ‘CampSessions’ table, there are transitive dependencies such as the ‘dateRange’ and ‘sessionTime’ on the session number (i.e., 1 through 20). However, the actual values of the ‘dateRange’ attribute, as alluded to in Challenge 2, changes from year-to-year. Thus, we decided that even though redundancy is likely to be increased, improved understandability would similarly increase by not aggressively pursuing 3NF here.

(3) There is possible transitive dependency of emergency treatment (‘emergencyTx’) on a camper’s medical condition (‘medCondition’), but because treatment for the same medical condition (e.g., asthma) can vary between individuals we did not consider this to be a true case of transitive dependency**.**