UBLIS 571 Soergel 2012Sp

**Lecture 4.2 Conceptual data schema exercise Step 1: From anticipated questions to conceptual data schema draft**

In the questions column, I omitted the listing of the entity types by themselves ‒ they can be seen from the relationship types. Entity types and relationship types not already listed for a previous question underlined. The underlined entity types and relationship types were copied to their respective columns to give a very rough draft of the conceptual data schema, the material from which the final schema can be built. This is a first step of distillation.

|  |  |  |
| --- | --- | --- |
| **Questions** | **Entity types** | **Relationship types** |
| **Who knows about printers?**  Person <*knowsAbout*> Subject  **I am looking for a review of Microsoft Word 12**  Document <*reviews*> SoftwareMake&Model  **What is a good word processor for Red Hat Linux**  SoftwareMake&Model <*servesFunction*> Function  SoftwareMake&Model <*worksWith***>** SoftwareMake&Model  SoftwareMake&Model <*hasQuality>* QualityScore  SoftwareMake&Model <*hasAttribute>* Quality  **Where will the Red Hat Linux group meet?**  Note: Question is underspecified. Should be When and where  UserGroup <*meetsIn*> (DateTime, Location  UserGroup <*takesplacein*> (DateTime, Location)  **When is the meeting for the Red Hat Linux user group?**  MeetingDate, UserGroup, SoftwareMake&Model  UserGroup <*dealsWith>* SoftwareMake&Model  UseGroup <*meetsOn>* MeetingDate  Note: MeetingDate is not a good entity type because it includes a role, date of a meeting. DateTime is a good entity type (plain date is DateTime given with low precision).  **When does the IT Club meet?**  Club <*meetsOn*> MeetingDate  Note: Same as previous.  **When does the next computer group meeting take place and whom can I call for more information?**  user group *<meetsOn>* meeting date  person *<hasKnowledgeOf>* subject  person *<contactBy>* phone number  **Who will write the newsletter?**  Newsletter <*authoredBy*> Person  Person <*isAuthorOf*> Newsletter  **What is the phone number to get help with word processing?**  Person <*knowsAbout>* Subject  Person <*contactBy>* PhoneNumber  **Who wrote the product review on MS Word?**  Document <*reviews>* SoftwareMake&Model  Document <*authoredBy>* Person  **Where can I get the best price for the printer I selected?**  HardwareMake&Model <*hasPrice*> (Organization, MoneyAmount)  **What is the phone number of the company I found?**  Organization <*hasPhoneNumber*> PhoneNumber  **Is there a fee to join the club?**  Note: Note a good question to ask of the club's database. Look in the club's bylaws  Person <isMemberof> Club  Club <hasFeeof> Cost  Cost? <isPaidby> Person  **Who is in the computer club?**  Club <*hasMember*> Person  **How can members be contacted?**  Person <*isMemberof*> Club  Person <*hasEmail*> Contact  **What articles have been published by members?**  Person <*has Published*> Document  Note: Means <*has Authored*>, which is the same as <*AuthorOf*>  Document <*authoredBy>* Person  **What graphics cards are available for the Lenovo W510?**  HardwareMake&Model <*worksWith*> HardwareMake&Model  **Does SimCity work with the Invidia 700 graphics card?**  SoftwareMake&Model <*worksWith*> HardwareMake&Model  **What member can set up a computer network?**  Person <*hasKnowledge*> Subject  Note: Already covered as <*knowsAbout*>  **What subjects are the experts knowledgeable in?**  Person <*hasKnowledgeOf*> Subject  Note: Curiously phrased. Should say What subject is expert A knowledgeable? in OR Who knows about a given subject?  **I'm searching for an article about creating websites.**  Document <*dealsWith*> Subject  **Who is the creator of the Website?**  Person <*creatorOf*> Object  Note: Object seems too general for Website. Should use Website for now and think about this when reorganizing the initial schema.  **Where is the Chairpersons office?**  Person <*locatedAt*> Location  Note: How do you know who is the chair person. Need Person <*servesInRole*> (PersonRole, Organization) A role is always in the context of an Organization, so this needs a three-way relationship.  **Who is the contact person for the group purchasing program?**  user group <*carriesOut*> program  person <*affiliatedWith*> program  Note: <*servesInRole*> is better here; the role is contact person.  person <*contactBy*> phone number`  **Who can fix my computer, how do I contact them?**  Person <knowsAbout> Subject  Person <contactBy> Phone Number  Note: This could als be an organization, such as a computer store or computer repair company, so we also need Organization <*knowsAbout*> Subject  Organization <*contactBy*> Phone Number  **`Who services the printer and how can we contact for service?**  Note: What printer?  HardwareMake&Model <*servesFunction*> Function  Company <*knowsAbout*> Subject  Company <*contactBy>* Email  Note: Organization is the better entity type. Also need generic entity types ServiceType and ServiceEvent Organization <*performs*> (ServiceType, HardwareMake&Model, MoneyAmount)    **How much will repair cost?**  Note: What repair?  Repair <*hasFeeof*> Cost  Note: Same question, What repair? What would a statement in the database look like? What are the values of the entity type Repair? Organization <*performs*> (ServiceType, HardwareMake&Model, MoneyAmount) where ServiceType would be Repair (as opposed to, for example, maintenance); in real life it would be much more specific: the specific repair for this specific printer model.  **Where do I find an article about fixing a computer?**  Document <*dealsWithSubject*> Subject  **What is the best printer for printing photos?**  Hardware <*bestFor*> Function  Note: Instead use HardwareMake&Model <*hasQuality*> (Function, QualityScore>, an extension of the two-way relationship <*hasQuality*>; the system can then find the HardwareMake&Model with the best score.  **What is the best anti-virus program for a Windows PC?**  Software <*doesFunction*> Function  Software <*worksWith*> Hardware  Software <*bestQuality*> Quality  Note: It needs to be SoftwareMake&Model. Also, the <*worksWith*> relationship needs to be with the operating system, not the hardware. <*bestQuality*> treated as in the previous question  **What is the phone number to get help with word processing?**  Person <knowsAbout> Subject  Person <contactBy> PhoneNumber  **Whom do I contact for IT support?**  Person <*hasKnowledgeOf*> Subject  Person <*contactBy*> Phone number  **What is the phone number for canceled meetings?**  Person, PhoneNumber, Subject  Person <*knowsAbout>* Subject  Person <*contactBy>* PhoneNumber | Person  Subject  Document  SoftwareMake&Model  Function  QualityScore  UserGroup  DateTime  Location  Newsletter  PhoneNumber  Organization  MoneyNumber  Club  EmailAddress  HardwareMake&Model  Website  PersonRole  Organization  Program  ServiceType  ServiceEvent | Person <*knowsAbout*> Subject  Document <*reviews*> SoftwareMake&Model  SoftwareMake&Model <*servesFunction*> Function  SoftwareMake&Model <*worksWith*> SoftwareMake&Model  SoftwareMake&Model <*hasQuality>* QualityScore  SoftwareMake&Model <*hasAttribute>* (Quality, QualityScore)  Note: <hasQuality*>* is more specific than *<hasAttribute>* and may not be needed  UserGroup <*meetsIn*> (DateTime, Location  UserGroup <*takesplacein*> (DateTime, Location)  Note: May not need both relationships  Newsletter <*authoredBy*> Person  Person <*isAuthorOf*> Newsletter  Note: <isAuthorOf> is the reciprocal of <*authoredBy*> *;* need only one  Person <*contactBy>* PhoneNumber  HardwareMake&Model <*hasPrice*> (Organization, MoneyNumber)  Organization <*hasPhoneNumber*> PhoneNumber  Club <*hasMember*> Person  Person <isMemberof> Club  Note: <hasMember>is reciprocal of <*isMemberof*>; only one is needed.  Person <hasEmail> Contact  Note: Contact is a poor entity type; what would be its values? Should be EmailAddress  HardwareMake&Model <*worksWith*> HardwareMake&Model  SoftwareMake&Model <*worksWith*> HardwareMake&Model  Document <*dealsWith*> Subject  Person <*creatorOf*> Website  Person <*servesInRole*> (PersonRole, Organization)  Person <*locatedAt*> Location  UserGroup <carriesOut> Program  Organization <*knowsAbout*> Subject  Organization <*contactBy*> Phone Number  HardwareMake&Model <*servesFunction*> Function  Organization <*performs*> (ServiceType, HardwareMake&Model, MoneyAmount)  HardwareMake&Model <*hasQuality*> (Function, QualityScore> |

UBLIS 571 Soergel 2012Sp

**Lecture 4.2 Conceptual data schema exercise Step 2: Refining the conceptual data schema**

This steps starts from the raw entity types and relationship types that were distilled from the original questions; there is no need to look at the many original questions any more.

In this step, one first arranges the draft relationship types into groups that address similar types of data. The groups are indicated by color in the draft relationship type column and listed in the scratchpad column. Now one can see relationship types that are the same or almost the same and can be consolidated. One also sees patterns that suggest how the conceptual data schema can be simplified. There are many other types of editing. The results are seen in the column Relationship types, final. I first extracted the purple group and dealt with it, then the red group, etc. The audio gives more explanation.

HardwareMake&Model is an entity type for such values as Lenovo (Make) W510 (Model). When necessary, it is abbreviated as HardwareM&M. Same for SoftwareMake&Model and ProductMake&Model

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Draft conceptual data schema** | | **Final conceptual data schema** | | |
| **Entity types** | **Relationship types** | **Entity types** | **Relationship type, scratchpad** | **Relationship types, final** |
| Person  Subject  Document  SoftwareMake&Model  Function  QualityScore  UserGroup  DateTime  Location  Newsletter  PhoneNumber  Organization  MoneyNumber  Club  EmailAddress  HardwareMake&Model  Website  PersonRole  Organization  Program  ServiceType  ServiceEvent | Person <*knowsAbout*> Subject  Document <*reviews*> SoftwareMake&Model  SoftwareMake&Model <*servesFunction*> Function  SoftwareMake&Model <*worksWith*> SoftwareMake&Model  SoftwareMake&Model <*hasQuality>* QualityScore  SoftwareMake&Model <*hasAttribute>* (Quality, QualityScore)  Note: <hasQuality*>* is more specific than *<hasAttribute>* and may not be needed  UserGroup <*meetsIn*> (DateTime, Location  UserGroup <*takesplacein*> (DateTime, Location)  Note: May not need both relationships  Newsletter <*authoredBy*> Person  Person <*isAuthorOf*> Newsletter  Note: <isAuthorOf> is the reciprocal of <*authoredBy*> *;* need only one  Person <*contactBy>* PhoneNumber  HardwareMake&Model <*hasPrice*> (Organization, MoneyNumber)  Organization <*hasPhoneNumber*> PhoneNumber  Club <*hasMember*> Person  Person <isMemberof> Club  Note: <hasMember>is reciprocal of <*isMemberof*>; only one is needed.  Person <hasEmail> Contact  Note: Contact is a poor entity type; what would be its values? Should be EmailAddress  HardwareMake&Model <*worksWith*> HardwareMake&Model  SoftwareMake&Model <*worksWith*> HardwareMake&Model  Document <*dealsWith*> Subject  Person <*creatorOf*> Website  Person <*servesInRole*> (PersonRole, Organization)  Person <*locatedAt*> Location  UserGroup <carriesOut> Program  Organization <*knowsAbout*> Subject  Organization <*contactBy*> Phone Number  HardwareMake&Model <*servesFunction*> Function  Organization <*performs*> (ServiceType, HardwareMake&Model, MoneyAmount)  HardwareMake&Model <*hasQuality*> (Function, QualityScore>  > | LegalEntity  Person  Organization  Subject  Document  DocumentM&M  DocumentType  PhoneNumber  EmailAddress  ProductMake&Model  SoftwareM&M  HardwareM&M  DocumentM&M  Function  Attribute  AttributeMeasurement  MoneyAmount  ServiceType  ServiceEvent  Event  EventType  DateTime  Location  LegalEntityRole  Program | Purple  Person <*knowsAbout*> Subject  Person <isC*reatorOf*> Entity  Person <*isAuthorOf*> Document  ~~Person <~~*~~creatorOf~~*~~> Website~~  ~~Person <~~*~~contactBy>~~* ~~PhoneNumber~~  ~~Organization <~~*~~contactBy~~*~~> Phone Number~~  Person <*hasPhoneNumber>* PhoneNumber  Organization <*hasPhoneNumber*> PhoneNumber  ~~Person <hasEmail> Contact~~  Person <hasEmail> EmailAddress  Organization <*knowsAbout*> Subject  Red  SoftwareMake&Model <*servesFunction*> Function  HardwareMake&Model <*servesFunction*> Function  SoftwareMake&Model <*worksWith*> SoftwareMake&Model  HardwareMake&Model <*worksWith*> HardwareM&M  SoftwareMake&Model <*worksWith*> HardwareM&M  SoftwareMake&Model <*hasQuality>* QualityScore  SoftwareMake&Model <*hasAttribute>* (Quality, QualityScore)  Note: <hasQuality*>* is more specific than *<hasAttribute>* and may not be needed  HardwareMake&Model <*hasQuality*> (Function, QualityScore>  HardwareM&M <*hasPrice*> (Organization, MoneyNumber)  Organization <*performs*> (ServiceType, HardwareMake&Model, MoneyAmount)  Blue  Document <*reviews*> SoftwareMake&Model  Document <*dealsWith*> Subject  Green  UserGroup <*meetsIn*> (DateTime, Location  UserGroup <*takesplacein*> (DateTime, Location)  Note: May not need both relationships  Club <*hasMember*> Person  Person <isMemberof> Club  Note: <hasMember>is reciprocal of <*isMemberof*>; only one is needed.  Person <*servesInRole*> (PersonRole, Organization)  Person <*locatedAt*> Location  UserGroup <carriesOut> Program | LegalEntity <*knowsAbout*> Subject  LegalEntity <isC*reatorOf*> Entity  LegalEntity <*isAuthorOf*> Document  LegalEntity <*hasPhoneNumber>* PhoneNumber  LegalEntity <hasEmailAddress> EmailAddress  ProductMake&Model <*servesFunction*> Function  ProductMake&Model <*worksWith*> ProductM&M  ProductMake&Model < *hasAttribute*>   (Attribute, Function, AttributeMeasurement)  ProductM&M <*hasPrice*>   (Organization, MoneyAmount)  LegalEntity <*performs*>   (ServiceType, ProductM&M, MoneyAmount)  Document <*dealsWith*> Subject  Document <*reviews*> Entity  (LegalEntity-1, LE-2, LE-3, . . .) <*participateIn>* Event  Event <*isa*> EventType  Event <*takesPlaceIn*> (DateTime, Location)  LegalEntity <*isMemberof*> Organization  LegalEntity <*servesInRole*>   (LegalEntityRole, LegalEntity)  LegalEntity <*locatedAt*> Location  LegalEntity <carriesOut> Program |