Edinburgh Global Game Jam

Jam year: 2018

#EDINBURGHGGJ

Games Lab (B56), Merchiston Campus
Edinburgh Napier University, 10 Colinton Road, Edinburgh, EH10 5DT, United Kingdom

Notes:
5:00pm to 6:00pm - Arrival, Room B32
(if you get here early, you can chill in the cafe)
6:15 - Keynote, Theme announcement.
6:45 - Group Forming
7:15pm - Jam Starts.

Sunday
3-5pm - Playtests
4pm - Submissions close

Jam Site Organizers

Attenborough

dooglz
Games Lab (B56), Merchiston Campus
Edinburgh Napier University, 10 Colinton Road, Edinburgh, Eh10 5DT, United Kingdom

Notes:
5:00pm to 6:00pm - Arrival, Room B32
(if you get here early, you can chill in the cafe)
6:15 - Keynote, Theme announcement.
6:45 - Group Forming
7:15pm - Jam Starts.

Sunday
3-5pm - Playtests
4pm - Submissions close

Website: http://games.soc.napier.ac.uk/ggj/

Entrance Fee: None

Hours: Open for entire 48 hours
Homework from Tuesday

Read This

Before the next lecture, on Friday, read the remaining sections, §§2.5 onwards, of Ramakrishnan and Gehrke, completing Chapter 2.

These consider trade-offs and choices in the design of Entity-Relationship models, as well as more on the wider context of modelling.

Data Representation

This first course section starts by presenting two common data representation models.

- The entity-relationship (ER) model
- The relational model

Data Manipulation

This is followed by some methods for manipulating data in the relational model and using it to extract information.

- Relational algebra
- The tuple-relational calculus
- The query language SQL
The tutorial worksheet was posted to the course website earlier this week. Download it, read the instructions, follow them.

The sheet has six questions where you progressively design and then draw an Entity-Relationship model for part of a database system.

There is flexibility in how you design the model, and there is more than one possible solution to the problem.

If you find parts difficult, or have questions about the exercise: ask other students; ask on Piazza; drop in to InfBASE; or bring your problem to the tutorials.

Each tutorial worksheet also has several more example problems and notes on solving them. Have a look at these, and their solutions, but you don’t have to do them for the tutorials.
How to Tutorial

Before the tutorial, work through the exercise sheet. Ask and answer questions on Piazza, talk to other students, write up your work.

Bring your solutions and your working. You will need to be able to show these to your tutor and exchange them with other students.

Come to tutorials properly prepared. Students who have not attempted the exercises will be sent away to spend the time working on them.

Data & Analysis tutorials and exercises are not given marks or grades, but their content is examinable and they are an important part of the course. If you do not do the exercises then you are unlikely to pass the exam.

Course participation requires attendance at tutorials: if you are ill or otherwise unable to attend one week then email your tutor, and if possible attend another tutorial group in the same week.
Remote Working

You can access many DICE services and Informatics resources remotely.

- Files over the web. https://ifile.inf.ed.ac.uk
- Command line. ssh student.ssh.inf.ed.ac.uk then ssh student.login
  (On Microsoft Windows, use PuTTY to reach student.ssh.inf.ed.ac.uk)
- Desktop. Graphical login with NX to nx.inf.ed.ac.uk

You can also use Virtual DICE, tunnel X Windows, access files over AFS, and connect by VPN to internal networks at Informatics and the University.

http://computing.help.inf.ed.ac.uk
The Story so Far

Entity-Relationship diagrams

- Entities: Rectangles
- Relationships: Diamonds linked to the entities
- Attributes: Ovals linked to entity or relationship

Entity-Relationship diagram:

- Student
  - Attributes: UUN, Age, email
- Takes
- Course
  - Attributes: Code, Title, Year
- Mark
  - Attributes: Mark, Title, Code

Key constraints as arrows

Weak entities with their identifying relationship
The Story so Far

Entity-Relationship diagrams

- Entities: Rectangles
- Relationships: Diamonds linked to the entities
- Attributes: Ovals linked to entity or relationship
- Key constraints as arrows

```
Container  On  Ship
```

Weak entities with their identifying relationship
The Story so Far

Entity-Relationship diagrams

- Entities: Rectangles
- Relationships: Diamonds linked to the entities
- Attributes: Ovals linked to entity or relationship
- Key constraints as arrows; total participation as thick/double lines

Diagram:

```
Country -- Uses --> Language
```
The Story so Far

Entity-Relationship diagrams
- Entities: Rectangles
- Relationships: Diamonds linked to the entities
- Attributes: Ovals linked to entity or relationship
- Key constraints as arrows; total participation as thick/double lines

Diagram:

- Course
- Has
- Exam
Entity-Relationship diagrams

- **Entities:** Rectangles
- **Relationships:** Diamonds linked to the entities
- **Attributes:** Ovals linked to entity or relationship
- **Key constraints as arrows; total participation as thick/double lines**
- **Weak entities with their identifying relationship**
Entity-Relationship diagrams

- Entities: Rectangles
- Relationships: Diamonds linked to the entities
- Attributes: Ovals linked to entity or relationship
- Key constraints as arrows; total participation as thick/double lines
- Weak entities with their identifying relationship; Entity hierarchies
The Story so Far

Entity-Relationship diagrams
- Entities: Rectangles
- Relationships: Diamonds linked to the entities
- Attributes: Ovals linked to entity or relationship
- Key constraints as arrows; total participation as thick/double lines
- Weak entities with their identifying relationship; Entity hierarchies

Relational models
- Relations: Tables matching schemas
- Schema: A set of field names and their domains
- Table: A set of tuples of values matching these fields
- Primary key: Chosen uniquely-valued field or set of fields
- Foreign key: Must be drawn from key values in another table
The Story so Far

Schema

Fields (a.k.a. attributes, columns)

<table>
<thead>
<tr>
<th>mn</th>
<th>name</th>
<th>age</th>
<th>email</th>
</tr>
</thead>
<tbody>
<tr>
<td>s0456782</td>
<td>John</td>
<td>18</td>
<td>john@inf</td>
</tr>
<tr>
<td>s0412375</td>
<td>Mary</td>
<td>18</td>
<td>mary@inf</td>
</tr>
<tr>
<td>s0378435</td>
<td>Helen</td>
<td>20</td>
<td>helen@phys</td>
</tr>
<tr>
<td>s0189034</td>
<td>Peter</td>
<td>22</td>
<td>peter@math</td>
</tr>
</tbody>
</table>

Relational models

- Relations: Tables matching schemas
- Schema: A set of field names and their domains
- Table: A set of tuples of values matching these fields
- Primary key: Chosen uniquely-valued field or set of fields
- Foreign key: Must be drawn from key values in another table

Ian Stark
Inf1-DA / Lecture 4
2018-01-26
CREATE TABLE Takes ( uun VARCHAR(8),
  code VARCHAR(20),
  mark INTEGER,
  PRIMARY KEY (uun, code),
  FOREIGN KEY (uun) REFERENCES Student,
  FOREIGN KEY (code) REFERENCES Course )

Relational models

- Relations: Tables matching schemas
- Schema: A set of field names and their domains
- Table: A set of tuples of values matching these fields
- Primary key: Chosen uniquely-valued field or set of fields
- Foreign key: Must be drawn from key values in another table
Aim and Scale

The University of Edinburgh has 39,756 matriculated students studying 7,743 different courses across 20 different schools in 726 buildings.

With a relational database we can ask, and answer, questions like:

- What are the names and email addresses of all students taking a first-year course taught by the School of Informatics?
- Which lectures have been scheduled for rooms which will be at greater than 90% of their seating capacity?

Many databases are much, much larger than this with much more complex queries. However, the aim is the same: how to go beyond answering a single query with a single program to a general system in which many queries can be expressed and answered efficiently.
An ER diagram captures a conceptual model of the data to be managed in a database: what there is and how it is connected.

We can use this as a basis for a relational schema, as a step towards implementation in a working RDBMS (Relational Database Management System).

This translation will be approximate: some constraints expressed in an ER diagram might not naturally fit into relational schemas.

CREATE TABLE Student ( ... )
CREATE TABLE Takes ( ... )
CREATE TABLE Course ( ... )

etc.
There may be more than one possible translation: different alternatives lead to different implementations. These may be efficiency trade-offs, for which we might go back to the requirements to assess their relative impact.

It is possible to make these translations complete (work for any diagram) and automatic (in a push-button tool); but here we shall just consider a few examples illustrating some of the main ideas.

CREATE TABLE Student ( ... )
CREATE TABLE Takes ( ... )
CREATE TABLE Course ( ... )

/etc./
Create a table for the entity.

Make each attribute of the entity a field of the table, with an appropriate type.

Declare the field or fields which make up the primary key.

```sql
CREATE TABLE Student (  
uun VARCHAR(8),  
name VARCHAR(20),  
age INTEGER,  
email VARCHAR(25),  
PRIMARY KEY (uun) )
```
Create tables for each entity, and an additional table for the relationship.
Add all key attributes of all participating entities as fields in the relationship table.
Add further fields for any attributes of the relationship itself.
Declare primary key using all key attributes from participating entities.
Declare foreign key constraints for all fields in the relationship table that refer to attributes of the entities.
CREATE TABLE Takes (  
uun VARCHAR(8), year INTEGER,  
code VARCHAR(20), mark INTEGER,  
PRIMARY KEY (uun, code, year),  
FOREIGN KEY (uun) REFERENCES Student,  
FOREIGN KEY (code,year) REFERENCES Course )
Create tables for each entity, and an additional table for the relationship.

Add all key attributes of all participating entities as fields in the relationship table.

Add further fields for any attributes of the relationship itself.

Declare primary key using only key attributes of the source entity.

Declare foreign key constraints for all fields in the relationship table that refer to attributes of the entities.
CREATE TABLE Joins ( 
  uun VARCHAR(8),
  time VARCHAR(20), place VARCHAR(32),
  PRIMARY KEY (uun),
  FOREIGN KEY (uun) REFERENCES Student,
  FOREIGN KEY (time,place) REFERENCES InfPALSGroup )
In fact, because the Joins relationship is many-to-one, we don’t need a whole table for the relationship itself: information about InfPALS group membership can go in the Student table.

- Create tables for the source and target entities as usual.
- Add every key attribute of the target as a field in the source table.
- Declare these fields as foreign keys.
CREATE TABLE Student ( 
    uun VARCHAR(8), age INTEGER, 
    name VARCHAR(20), email VARCHAR(25), 
    time VARCHAR(10), place VARCHAR(32), 
    PRIMARY KEY (uun), 
    FOREIGN KEY (time,place) REFERENCES InfPALSGroup )
Null Values

A field in SQL can have the special value **NULL**.

**NULL** can mean many things: that a field is unknown, or missing, or unavailable; or that this field may not apply in certain situations.

Some RDBMS forbid **NULL** from appearing in any field declared as a primary key.

Some of these may still allow **NULL** to appear in foreign key fields.

A schema can state that certain fields may not contain **NULL** using the **NOT NULL** declaration.

Forbidding **NULL** is in some cases a way to enforce total participation.
Create tables for the source and target entities as usual.

Add every key attribute of the target as a field in the source table.

Declare these fields as **NOT NULL**

Declare these fields as foreign keys.
CREATE TABLE Student ( 
  uun VARCHAR(8), age INTEGER,  
  name VARCHAR(20), email VARCHAR(25),  
  pt VARCHAR(8) NOT NULL,  
  PRIMARY KEY (uun),  
  FOREIGN KEY (pt) REFERENCES Staff(staff_id) )
A weak entity always has a participation and key constraint with its identifying relationship, which makes things similar to the previous case.

- Create a tables for the weak entity and its identifying owner.
- Add every key attribute of the identifying owner as a field in the weak entity table.
- Declare a composite key using key attributes from both entities.
- Declare a foreign key constraint on the identifying owner fields.
- Declare that records in the table must be deleted when their identifying owners are.
CREATE TABLE Room (  
  number VARCHAR(8),  
  building_name VARCHAR(20),  
  capacity INTEGER,  
  PRIMARY KEY (number, building_name),  
  FOREIGN KEY (building_name) REFERENCES Building(name)  
    ON DELETE CASCADE )

(Don’t use ON DELETE SET NULL here)
• Declare a table for the superclass entity with all attributes.
• For each subclass entity declare another table using the primary key of the superclass and any extra attributes of the subclass.
• Declare the primary key from the superclass as the primary key of each subclass, with a foreign key constraint.
CREATE TABLE Student (  
  uun VARCHAR(8), age INTEGER,  
  name VARCHAR(20), email VARCHAR(25),  
  PRIMARY KEY (uun) )
CREATE TABLE FullTimeStudent (
    uun  VARCHAR(8),
    PRIMARY KEY (uun),
    FOREIGN KEY (uun) REFERENCES Student
)
CREATE TABLE PartTimeStudent ( 
  uun VARCHAR(8), fraction FLOAT, 
  PRIMARY KEY (uun), 
  FOREIGN KEY (uun) REFERENCES Student )
Database Textbooks (Page 1 of 2)

R. Ramakrishnan and J. Gehrke.  
*Database Management Systems.*  
3 copies in HUB  
6 other copies

H. Garcia-Molina, J. Ullman, and J. Widom.  
1 copy in HUB

J. Ullman and J. Widom.  
*A First Course in Database Systems.*  
1 copy in HUB  
1 other copy

M. Kifer, A. Bernstein, and P. M. Lewis.  
1 copy in HUB
A. Silberschatz, H. Korth, and S. Sudarshan.  
*Database System Concepts.*  

T. Connolly and C. Begg.  

R. Elmasri and S. Navathe.  
*Fundamentals of Database Systems.*  
Conceptual Model

ER-diagrams: entities with attributes and the relationships between them.

Logical Model

Relational models: tables with fields that can refer to other tables.

Translation

We can use an ER-diagram to guide the design of an appropriate relational model. This may require choosing between alternatives and making compromises: possibly involving simplicity of the design or its efficient implementation.
Homework

Do This

Find on the course website the worksheet with the first set of tutorial exercises.

Download it, read the instructions on the front page, follow them.