Databases

A **database** supports storage and retrieval of **structured** data. The data are organized into a set of **tables,** and those tables can be **joined** together using keys. In each table, the **rows** (also known as records) each represent a single member of that entity (examples: a movie, a person, an address, an invoice, a customer account). The **columns** (also known as fields or attributes) represent one piece of data for that entity, with a particular type. A column can be a string of text, a number, a date/time, an IP address, etc.

A **primary key** is an attribute that **uniquely identifies** each row in a table. That is, there can be no duplicates. Often the best thing to use as a primary key is an arbitrary integer. A good example of this is your student ID from LIU. It's possible, but a bit tricky, to user other, non-arbitrary data as a primary key. For example, if we were to use a customer's first and last name, there could be duplicates. So to make it a legitimate primary key, we'd have to sometimes add a number to ensure uniqueness. LIU does this with your logins (such as `jane.smith4`) and IMDB uses Roman numerals to distinguish people with the same name – such as Emma Watson (II) vs. Emma Watson (I). Other examples of integer primary keys are your passport number, driver's license number, social security number, and account numbers with other organizations.

A **foreign key** is an attribute that references the primary key of a **different table.** It is through the use of foreign keys that we can join together multiple tables in a database. The handout below illustrates this. The first two tables (Movie and Person) have integer primary keys in the "ID" column. Then third table references those primary keys. To answer some of the questions below, you have to look into all three tables.

- Handout with a sample database of artists, museums, and paintings
- SQL statements to answer the questions on the artist database handout
- Sample SQL statements for a movie database