FIT2094-FIT3171 2019 S1 -- Week 3 eBook

Credits and Copyrights:
Authors: FIT3171 2018 S2 UG Databases
FIT Database Teaching Team
Maria Indrawan, Manoj Kathpalia, Lindsay Smith, et al

Change Log:

- Formatted and Imported from Alexandria March 2019.

3.0. The Relational Data Model

In this week's work, we will look at the fundamental concepts on which the Relational Database model is built.

3.0.1. Terminology

Discuss the following terms:

- Relation
- Attribute
- Domain
- Tuple
- Degree and Cardinality of a Relation

3.0.2. Scenario: Purchasing items in a modern online game

Scenario: you are working for Blizzard Entertainment to design a database for storing player orders for items (e.g. equipment, merchandise) in the World of Warcraft online game.

Consider the PLAYER and ORDER relations below:

- **PLAYER (PLAYER-ID, FULLNAME, EMAIL)**
- **ORDER (ORDER-ID, DATE, PLAYER-ID)**

Assume a single player may have any number of orders.

1. Identify the primary key and foreign key attributes in these relations.

2. Can you think of a reason why we would not just store all the player and order information in one relation, so that we would not have to perform the join operation?

3.0.3. Choosing Primary Keys

1. In any relation, **tuples must be unique**. However, in many cases, the set of all the attributes in a relation is **not** considered a candidate key. **Why not?**
   On the other hand, suppose we do have a relation where the set of all attributes is a candidate key. In this case, show that this set must, therefore, be the only candidate key and hence the primary key.
2. **Scenario: Tutorial Room Bookings at Monash.**
   Consider a relation that depicts a tutorial room booking in a university. Each faculty assigns a person to handle the booking for all tutorial classes for that faculty. The person's email address is given to the university's booking system as a contact person.
   BOOKING(b_date, b_starttime, b_endtime, unit_code, contact_person, room_no, tutor_id)
   a. Identify candidate key(s) and primary key for the relation if the following business rules are applicable:
      i. More than one tutorial classes of the same unit may run at the same time (parallel sessions are possible).
      ii. A tutor may teach several classes of the same unit.
      iii. All tutorial classes are 2 hours long.

b. Identify candidate key(s) and primary key for the relation if the following business rules are applicable:
    i. Tutorial classes can be either 1 hour or 2 hours long.
    ii. A tutor can only teach one tutorial class in a given unit.
    iii. There are no parallel sessions of tutorial classes.

3.0.4 Relational Algebra - Applications (Adapted from Exercise 3.6 of Connolly, Begg and Strachan)

Suppose we have the following 4 relations:

- **HOTEL(HOTEL-NO, NAME, ADDRESS)**
- **ROOM(ROOM-NO, HOTEL-NO, TYPE, PRICE)**
- **BOOKING(HOTEL-NO,GUEST-NO, DATE-FROM, DATE-TO, ROOM-NO)**
- **GUEST(GUEST-NO, NAME, ADDRESS)**

Generate the relational algebra for the following queries:

1. List the names and addresses of all hotels
2. List all single rooms with a price below $50
3. List the names and addresses of all guests
4. List the price and type of all rooms at the Grosvenor Hotel
5. List all names and addresses of guests currently staying at the Grosvenor Hotel (assume that if the guest has a tuple in the BOOKING relation, then they are currently staying in the hotel)

3.0.5 Relational Algebra - Theory

There are 7 relational algebra operators, namely:

***SELECTION*, *PROJECTION*, JOIN, *UNION*, INTERSECTION, *DIFFERENCE* and *CARTESIAN PRODUCT*.**

In fact, 5 of these operators may be considered primitive operators in the sense that the others may be expressed in terms of the primitive operators. **The primitive operators are:**

SELECTION, PROJECTION, UNION, DIFFERENCE and CARTESIAN PRODUCT (marked in italics in the first list).

Using the sample tables below, show how the JOIN operation can be expressed in terms of the fundamental operators by showing the process to do a **natural join** of CUSTOMER and ORDER.

- **CUSTOMER table:**

Cust_ID	Name
1	Green
2	Blue

- **ORDER table:**

Ord_ID	Date	Cust_ID
1	23-Feb-2009	1
2	26-Feb-2009	1
3	26-Feb-2009	2

3.0.6. Advanced Question (optional - only if time permits)

If your tutor has time at the end of the class, let's apply your knowledge from the world of relational algebra above to see if you can solve a theoretical problem.

The database below is for a department store and describes stock, staff, clients, and sales.

ITEM

ITEM	DESCRIP	PRICE
K3	Knife Set	$17.95
K5	Ladle	$6.95
K11	Scraper	$0.95
L12	Rack	$22.95
L3	Table	$399.50
L6	Stool	$17.95

SALE

ITEM	STAFF	NUMSOLD	CLIENT	DATE
K3	Simon	6	Clark	20170311
K11	Simon	1	Cilla	20170121
K11	Simon	1	Cilla	20170123
L12	Sorcha	5	Charles	20161130
K3	Sean	1	Clive	20170221
L12	Sean	1	Cilla	20170228

L12	Simon	2	Clive	20170228
K3	Sean	2	Charles	20161129

STAFF

<u>**NAME**</u>	**POSITION**
Sandra	Manager
Simon	Clerk
Steve	Packer
Sean	Clerk
Sorcha	Director
Sian	Clark

STOCK

<u>**ITEM**</u>	**NUMSTOCK**
K3	105
K11	66
L3	0
L12	4
L6	13

CLIENT

<u>**NAME**</u>	**ADDRESS**
Clive	India Rd
Clark	Kent St
Charles	Windsor Av
Cilla	Black St

Using Relational Algebra answer the following queries. You must represent your answer in symbolic notation and where a query has several solutions, your answer must represent the most efficient solution.

1. What items are out of stock?
2. What items haven't been sold?
3. Which clerks don't have any sales?
4. What categories (positions) of staff have made sales?

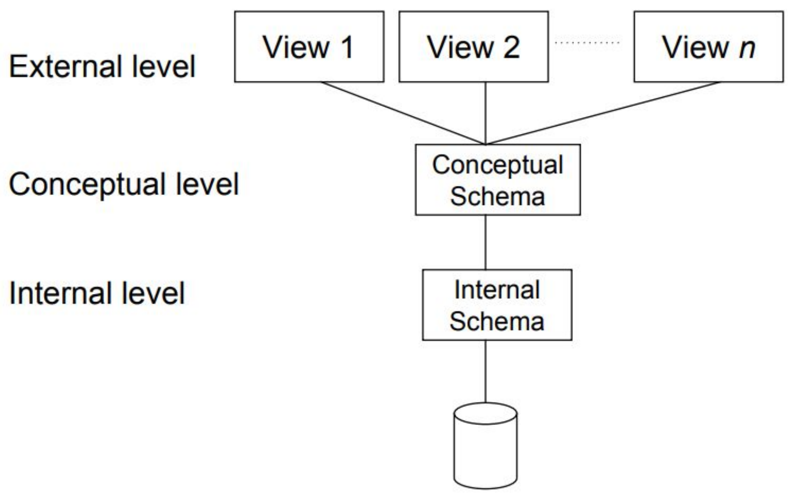
3.1. Pre-Lecture Notes

These are notes you may find useful to read through before the lecture, adapted from Lindsay Smith's lecture material. NOTE: THESE ARE NOT THE FINAL LECTURE SLIDES.

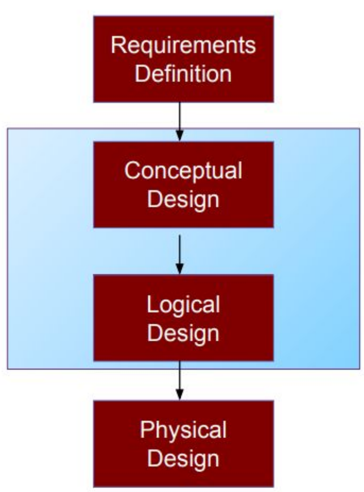
For a quick 2-minute read of Crow's Foot Notation with a case study for a gym's DBMS, refer Fey (2017): https://medium.com/@marcifey/using-crows-foot-notation-in-an-erd-2910fff5dd05

Read up the pre-lecture notes below:

ANSI/SPARC architecture



The Database Design Life Cycle



Requirements Definition

- Identify and analyse user views.
- A 'user view' may be a report to be produced or a particular type of transaction that should be supported.
- Corresponds to the external level of the ANSI/SPARC architecture.
- Output is a statement of specifications which describes the user views' particular requirements and constraints.

ER Modeling

- ER (Entity-Relationship) model developed by Peter Chen in 1976 to aid database design.
- May be used for conceptual (ERD)/logical design (ERD like).
- ER diagrams give a visual indication of the design.
- Basic components:
  - Entity
  - Attribute
  - Relationship

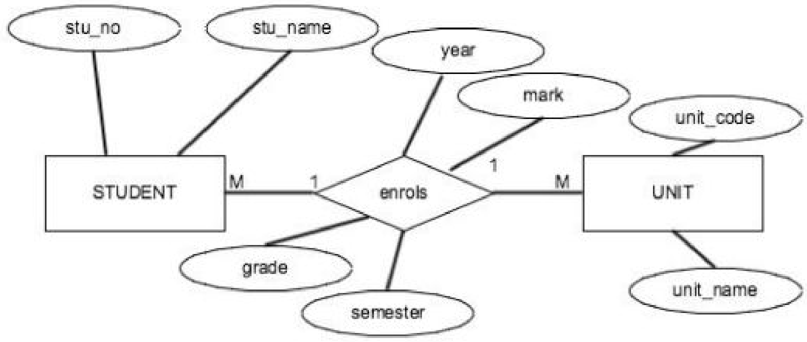
Logical Design

- Develop a data model which targets a particular database model (e.g. relational, hierarchical, network, object-oriented).
- Independent of any implementation details which are specific to any particular DBMS package.
- Normalisation technique (see week 5) is used to test the correctness of the logical model.
- May also be considered to correspond to the conceptual level of the ANSI/SPARC architecture.

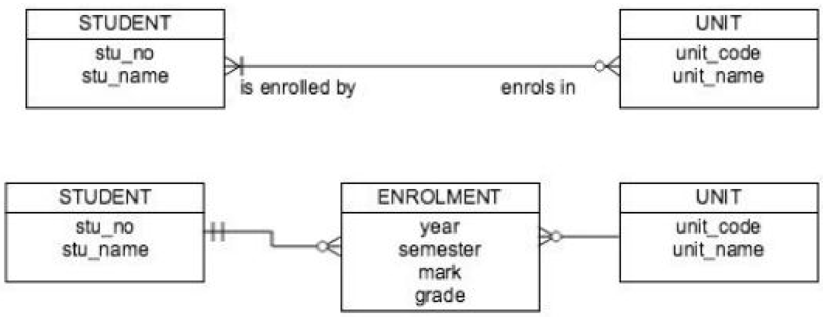
Physical Design

- Develop a strategy for the physical implementation of the logical data model.
- Choose appropriate storage structures, indexes, file organisations and access methods which will most efficiently support the user requirements.
- Physical design phase is dependent on the particular DBMS environment in use.
- ANSI/SPARC internal level.
- Shown in SQL Developer Data Modeller as the Relational Model

ERD - Notation



Chen



Information Engineering/James Martin/Crows foot

one to many

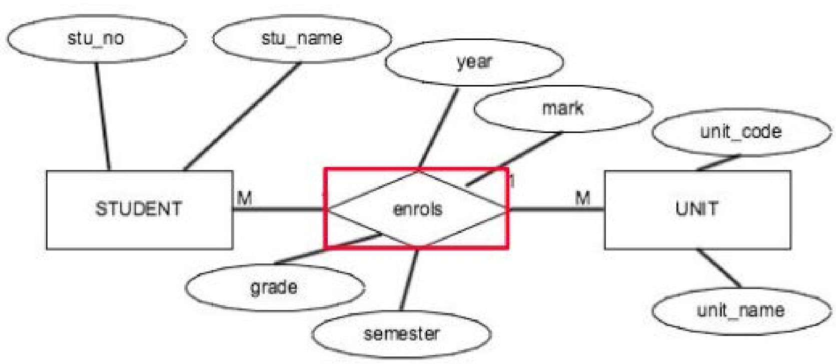
one to one

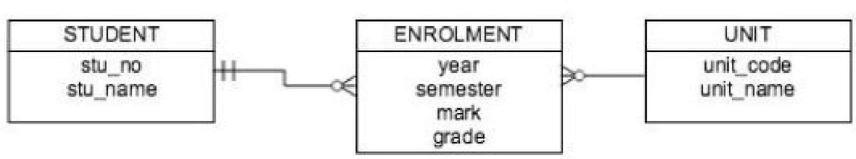
many to many

Types of Attributes

- Simple
  - Cannot be subdivided
  - Age, sex, marital status
- Composite
  - Can be subdivided into additional attributes
  - Address into street, city, zip
- Single-valued
  - Can have only a single value
  - Person has one social security number
- Multi-valued
  - Can have many values
  - Person may have several college degrees
- Derived
  - Can be derived with algorithm
  - Age can be derived from date of birth

Associative (or Composite) Entity





EOF.