CPSC 471 Notes

Brian Pho

June 22, 2017

Contents

1 Basic Concepts **2**

2 Entity Relationship Model **4**
- 2.1 Attributes . . . 4
  - 2.1.1 Types of Attributes . . . 4
  - 2.1.2 Key Attribute . . . 4
  - 2.1.3 Determining Key Attribute . . . 5
- 2.2 ER-Diagram Notation . . . 5
- 2.3 Entity . . . 5
  - 2.3.1 Weak Entity Type . . . 5
- 2.4 Relationship . . . 5
  - 2.4.1 Structural Contraints . . . 5
  - 2.4.2 Recursive Relationship . . . 6
  - 2.4.3 Attributes . . . 6
  - 2.4.4 Degree . . . 6

3 Extended Entity Relationship Model **7**
- 3.1 Abstractions . . . 7
- 3.2 EER Model Concepts . . . 7
- 3.3 Lattice Details . . . 8

4 Relational Data Model **9**
- 4.1 Definition . . . 9
- 4.2 Characteristics of Relations . . . 9
- 4.3 Relational Integrity Constraints . . . 9
  - 4.3.1 Key Constraints . . . 9
  - 4.3.2 Entity Integrity Constraint . . . 10
  - 4.3.3 Referential Integrity . . . 10
  - 4.3.4 Semantic Integrity Constraint . . . 10

5 Relational Algebra **11**

6 Functional Dependencies **12**

7 Normalization **13**

Chapter 1

Basic Concepts

Types of Databases

- Numeric and Textual
- Multimedia
- Geographic Information Systems
- Data Warehouses
- Real-time and Active

Basic Definitions

- Database (DB): A collection of related data
- Data: Known factions and information
- Mini-world: Some subset of the real world (context or frame of reference)
- Database Management System (DBMS): A system designed to manage databases
- Database System: The DBMS + Data

Main Characteristics of Databases

- Self-Describing Nature: A DBMS stores a description of itself which is called meta-data. This allows the DBMS to work with different databases.
- Insulation Between Programs and Data: Similar to how instructions and data are kept separate in caches.
- Data Abstraction: A data model used to hide storage details
- Support Multiple Views of the Data: User access level by hiding information
- Concurrency: Multiple users at the same time

Database Users

- Database Administrators: logistics and management of the DB
- Database Designers: creation and modification of the DB

- End-Users: users of the DB

Categories of End-Users

- Casual: Occasional access
- Naive/Parametric: Use of well-defined functions of the DB
- Sophisticated: Work more closely with the DB
- Stand-alone: Personal DB

Advantages of Using a Database

- Sharing of data among multiple users
- Efficient storage structures
- Multiple interfaces for different users
- Redundancy and ease of use

Disadvantages of Using a Database

- High initial investment and need for hardware
- Overhead for security, concurrency, recovery
- Hacking

NOSQL Database

- A non-relational and largely distributed database system that enables rapid, ad-hoc organization and analysis of extremely high-volume, disparate data types.
- Also known as cloud databases, Big Data databases, etc.

Chapter 2

Entity Relationship Model

- Also known as the ER Model
- Entities has attributes
- Attributes have name, domain, value
- All models are valid, but we are looking for the optimal solution
- Use composite attributes to make it easier to translate into a logical design
- Analogies: Class - Entity type, Object - Entity, Class data fields - attributes, Class methods - relationships

2.1 Attributes

2.1.1 Types of Attributes

- Simple - atomic attribute
- Composite - made up of other attributes
- Multi-Valued - multiple values for an attribute (same type)
- Calculated/Aggregated/Derived - **not stored** attribute that is calculated on the fly (dotted line circle). Not stored because it can be calculated every time

2.1.2 Key Attribute

- Every entity **must have** a key attribute which is unique to that entity
- A key attribute is used to differentiate between entities in the same entity type
- A key attribute may be composite
- An entity type may have more than one key attribute
- If a attribute of a composite attribute is a key attribute, then the composite attribute is also a key attribute
- Underline an attribute to show it is a key attribute

2.1.3 Determining Key Attribute

- If it's a unique signifier that's explicit in the question
- Assume some attribute is unique
- Select all possible key attributes if making an assumption

2.2 ER-Diagram Notation

- Multi-valued is a double lined circle (yellow outlines are double lined shapes)
- If a composite attribute is also a key attribute, underline the root attribute (the one closest to the entity)
- Can draw arrows on ER Diagrams to show relationship direction

2.3 Entity

2.3.1 Weak Entity Type

- Does not have a key attribute
- Dotted underline - weaker "key" attribute
- The weak key needs to borrow from it's identifying entity type

2.4 Relationship

2.4.1 Structural Contraints

Cardinality Constraints

- Many-to-One Relationship (N:1)
- One-to-Many Relationship (1:N)
- Many-to-Many Relationship (M,N)
- One-to-One Relationship (1:1)

Participation Contraints

- Total Participation - Both entities must contribute to the relationship
- Partial Participation - not total participation
- Double line for total, single line for partial
- Write assumptions if unclear with the diagram
- Any weak entity type must have total participation
- Must check participation for both ways

Min, Max Notation

- Know for tests
- Don't use participation constraints as it is already represented with numbers
- Location of numbers is opposite of cardinality

2.4.2 Recursive Relationship

- Shown as the line to itself
- A relationship where the entity acts on itself

2.4.3 Attributes

- Relationships act like entities that have attributes

2.4.4 Degree

- 2 - binary
- 3 - ternary
- n - n-ary

Chapter 3

Extended Entity Relationship Model

3.1 Abstractions

- Classification - single line arrow
- Aggregation - double line arrow
- Generalization -
- Specialization

Classification vs Generalization vs Aggregation

- In classification, all members of that class have the same attributes
- In generalization, they share attributes but don't have the same attributes
- in aggregation, a class is made up of subclasses

Coverage Constraints

- Partial coverage means some exception exists
- Disjointedness = Exclusive and Overlapping
- One-to-One means a function maps to exactly one value (E.g. $y = x$)
- Onto means a function maps to all values in a domain (E.g. $y = x$)

	Total	Partial
Disjoint	One-to-One and Onto ($y = x$)	One-to-One ($y = x$ but is discrete)
Overlapping	Not One-to-One and Onto ($y = x^2$)	Not One-to-One ($y = x^2$ but is discrete)

3.2 EER Model Concepts

- Specialization and generalization are two sides of the same coin

3.3 Lattice Details

Shared Subclass (multiple inheritance)

- When a subclass has more than one superclass
- A shared subclass gets its name from a class that is shared between one or more superclasses, hence it's called a subclass that is shared.
- **A shared subclass exists in all of its superclasses**
- In other words, the subclass must inherit ALL of its superclass' properties/attributes

Categories (Union Types)

- Similar to shared subclass but does not have to exist in all of its superclass' but at least one
- It's likes a math union with two sets overlapping. The subclass can be in either one, the other, or both. But it must be in at least one (aka it must exist somewhere within the two circles)

Chapter 4

Relational Data Model

4.1 Definition

- Scheme:
- Domain:
- Tuple:
- Cartesian Product:

4.2 Characteristics of Relations

- The order of the rows doesn't matter but the order of the columns matter
- The values in a tuple (attributes) are considered atomic

4.3 Relational Integrity Constraints

Constraints

- Key
- Entity Integrity
- Referential Integrity

4.3.1 Key Constraints

- Any set of attributes that can be used to identify any set of rows
- Must be unique for each row/tuple
- Can be any set which means you can use multiple columns to define the superkey (E.g. C_2, or $C_1 and C_3$)
- Key/Candidate Key: A "minimal" superkey; that is, a superkey K such that removal of any attribute from K results in a set of attributes that is not a superkey.

- To determine the candidate keys, look at the set of super keys, remove one of the columns from a key and see if it's a super key, it it is, then it is not a candidate key. Any super key with one attribute/column is an candidate key. A C.K. is a key where it's subset keys are not super keys.
- To determine super keys, a super key with a single attribute can be combined with other attributes to form more superkeys
- Example:
- Superkey List: [C2, C1 C3, C1 C2, C2 C3, C1 C2 C3]
- Candidate Key List: [C2, C1 C3]

4.3.2 Entity Integrity Constraint

4.3.3 Referential Integrity

- A constraint that deals with two or more tables/relations.
- It means that two tables must be consistent with their reference to each other
- E.g. Can't reference department ID 5 in employee table if ID 5 is not defined in the department table
- A foreign key is a primary key in another table

4.3.4 Semantic Integrity Constraint

Chapter 5

Relational Algebra

Two types of division: implicit and explicit (also known as dynamic and static). A static division is when you divide by the same items for all relations. Dynamic division is when you divide by different items for all relations. E.g. Find all employees who work in all projects in Calgary? (static) Find all employees who work ins all projects in their city?

Chapter 6

Functional Dependencies

Find F_{min} Algorithm

1. Apply decomposition rule to get only one attribute on RHS of every FD.
2. Eliminate redundant dependencies by finding X^+ for every $X \rightarrow Y$ in F. We need to find X^+ using all F except $X \rightarrow Y$. From this, if we can reach Y, that means $X \rightarrow Y$ is redundant and must be deleted.
3. Remove redundancy on LHS by deleting partial dependencies. So if we have $X \rightarrow Y$ and there is Z that is subset of X such that $Z \rightarrow Y$, then we have to replace $X \rightarrow Y$ with $Z \rightarrow Y$.

- $X \rightarrow Y$ is PFD (partially functionally dependent) if and only if there is Z subset from X, and $Z \rightarrow Y$.
- $X \rightarrow Y$ is FFD (fully functionally dependent) if and only if there is no subset in X such that this subset can determine Y.

Example: Given a relation R(A, B, C, D, E) and functional dependencies F($B \rightarrow D$, $AC \rightarrow BE$, $BD \rightarrow A$). Is A partially or fully functional dependent on BD?
Answer: A is PFD on BD as $B \subseteq BD$ can be used to reach A.

Example 2: Given R(A, B, C, D, M) and functional dependencies F($B \rightarrow AC$, $C \rightarrow A$, $DC \rightarrow MA$, $AD \rightarrow MC$). Find F_{min}.
Answer: First step is use decomposition rule

- $B \rightarrow A$
- $B \rightarrow C$
- $C \rightarrow A$
- $DC \rightarrow M$
- $DC \rightarrow A$
- $AD \rightarrow M$
- $AD \rightarrow C$

Second step is to eliminate redundant dependencies. $F_{min} = (B \rightarrow C, C \rightarrow A, AD \rightarrow M, AD \rightarrow C)$. Can check if F_{min} is equivalent to the original F by getting the closure of both sets. Get LHS from original F, then use the FD of F_{min}. The closure from using original F and F_{min} should be the same. Do vice versa. Closure is how much the FD can cover in the sets. Closure is calculated by A set including itself and then seeing where else it can lead to. 4 total cases.

Chapter 7

Normalization

Normalization is the breakdown of a given design into a more optimized equivalent design. Given a table R, an equivalent optimized design may break down R in $R_1, R_2, R_3, ..., R_n$. The two designs are equivalent if you can JOIN all of the smaller relations and still get the original R relation.

There are 4 normal forms: INF, 2NF, 3NF, BCNF. Any attribute not on the RHS of any FD must be part of CK (candidate key). Don't have to check supersets of a CK if you already found a CK. Closer of F must equal R for it to be a CK.