CSC 134 Notes

**DAY 1:**

**TOPIC: Database Management Systems**

* **Database**
  + **A database is a collection of related data.**
* **Database Management System (DBMS)**
  + **A database management system is a collection of programs that manipulate databases.**
* **Redundancy**
  + **Duplication, Waste storage, inconsistent**
* **Security**
  + **Beyond that provided by operating system**
  + **Authorized access**
* **Self- describing**
  + **Stores data, definition of database structure and constraints**
* **Integrity Constraints** 
  + **Data type, relationships, uniqueness**
* **Data abstraction**
  + **A change in a file system requires a change in the program.**
  + **A conceptual representation of data without details of physical storage**
* **Multiple views**
  + **A DBMS supports multiple users to view the database in different ways.**
  + **A view may a subset of the database, or contain virtual data derived from the database (not explicitly stored)**
* **Concurrency control**
* **Efficient query processing**

Database Users

* Database admin
* istrators (DBA)
  + Manage database and related software. E.g monitor use of resource, authorized access to the database, and system performance
* Database Designers
  + Design the structure to represent and store data
  + Identify the data to be stored
* End Users
  + Casual end user: use sophisticated database query language
  + Naïve user: menu-driven interface
  + Sophisticated end users: familiar with the DBMS to implement their applications

**Data Models**

Data model: a collection of concepts to describe the structure of a database

* High level/conceptual model
  + Entity-relationship model
* Representational/implementation model
  + Relation data model (class focus)
  + Object data model
  + Network model
  + Hierarchical model
* Low-level / physical data model
  + How data is stored in the disk

Day 2

Database languages:

* Data definition language (DDL)
  + use to specify database schema
* Data Manipulation Language (DML)
  + Manipulate and access data(e,g insert, retrieve change)

Database Management and file organization

* Entity – relationship Model
  + Conceptual Model
  + Entity, Relationship
  + ER diagram

Entity and Attribute

* Entity
  + An object with a physical existence or conceptual existence
  + E.g a person a company
  + Notation
  + Attribute is a property related to this
* Attributes
  + Properties and describe entities
  + E.g Name of an employee
  + Notation

Attribute

* Simple (atomic) attribute
  + Attribute that are ot divisable
  + E.g SSN,Zip code
  + Notation
* Composite Attribute
  + Can be devided into smaller subparts
  + E,g Address
  + Notation
* Stored Attribute
  + E.g birthdate
* Derived Attribute
  + Derived from other attribute
  + E.g age=current date-brithdate
  + Notation

Entity types and entity sets

* Entity Type
  + Defines a collection of entities that have the same attributes.
  + E.g employee
  + Describe the schema or intension for a set of entities that share the same structure
* Entity Set
  + The collection of all entities of a particular entity type in

Key

* A key is an attribute or the combination of multiple attributes that can be used to distinguish one entity instance from other entity instances in an entity type.
  + E.g ssn of an employee
* Composite key: a set of attributes as the key of an entity.
* Key must be minimal
* Notations
* Composite attributes as a key

Day 3

RELATIONSHIP

* Relationship type
* Defines a set of associations among entity types
* E.g employees work for a department
* Employee-----works\_for------Department
* Example of relationship instance

Relationship role name

* Each entity type that participates in a relationship type plays a particular role
* Role name: signify the rule that a participating entity from the entity type plays in each relationship instance
* E.g employee plays the role of worker department plays the role of employer

Cardinality ratios for binary relationship

* Epcify the number of relationshop instances that an entity can participate in.
* Possible cardinality ratios
  + 1:1 (one to one)
  + 1:N (one to many)
  + N:M (many to many)

Participation Constraints

* Total participation (existence depemdency)
  + Any employee must work for one department
* Partial participation
  + Some of the employee entities manage department entities, but not necessarily all.
  + Example

Weak Entity

* Figure
* Does not have key attributes of its own
* Has total participation constraints
* Partial Key: unique identifier of a weak entity that can be used to distinguish between weak entities related to the same owner entity

Attributes of relationship types

* An attribute conceptually belongs to the relationship
* M:N
  + Attributes may be determined by the combination of participating entities ina a relationship instance, not by any single entity
* 1:1
  + A relationship attribute can be migrated to one of the participating entity types
* 1:M
  + a relationship attribute can be migrated only to the entity type on the n-side
* 1:1 and 1:M determined subjectively by the schema designer

**Day 6:**

**Relational model concepts**

* Relational model presents a database a a collection of relations
  + Table :- relation
  + Row :- Tuple
  + Column header :- attribute
* **Name ssn home phone**
* Joe smith 307-88-2907 602-7765543
* Barbara Miller 590-38-6654 422-6655445

**Relational Model**

* Domain: a domain D in the relational model is a set of atomic values
  + Atomic: Each value in the domain is indivisible….

**Relational model**

* Relation schema
  + A relation schema R(a1,a2,an) is made up of a relation name r and a list of attributes a1,a2,an
    - E.g student (name,ssn,phonenumber)
  + Domain of AI is denoted by dom(AI) degree of a relation: number of attributes n of its relation schema.

**DAY 7:**

**Relational Model**

* Relation state
  + A relation (or relation state) of the relation schema R(a1,A2,….An) is denoted by r(R)
    - The relation is a set of m-tuples r={t1…tn}, where each n-tuple t is an ordered list of values: t=<v1,v2,..vn>
  + r(R) < (dom(A1) X dom (A2) X….X dom (An))
  + tuples are unordered in a relation
  + a relation cannot have duplicate tuples
  + denote cardinality (number of values) of domain D by |D|
  + maximum number of tuples in r(R) us |dom(A1)| \*|dom(A2)|\*…\*|dom(An)|
* Attribute value
  + Value Vi in tuple t for attribute Ai
    - T[Ai] for t.A
    - E.g. given tuple t =<’Joe Smith’,’307-88-2345’,’2342242525’>
      * T[name] =<’joe smith’>
* Catagory
  + Constraints on databases can generally be devided into three main catagories:
    - Inherent model based constraints
      * Constraints that are inherent in the data model
      * E.g.
        + Ordering of tuples in a relation
        + Relational model represents facts about both entities and relationship uniformly a relation
        + A relation cannot have duplicate tuples
    - Schema based constraints
      * Can be directly expressed in the schemas off the data model, typically ddl
      * Constraints are conditions that must hold on all valid relation states
      * Domain constraints
        + Within each tuple the value of each attribute a must be an atomic value from the domain dom(A)
        + Data type of domain

Integer

Boolean

…

* + - * Key constraints
        + SK is a super key of R, if for any two distinct tuples t1 and t2 in a relation state r of R, we have the constraint that t1[sk]++(the addition symbols are stacked) t2[sk]
        + Key constraint, unique constraint

No two distinct tuples in a any state r or R can have the same value for SK

* + - * + E.g. {SSN,Name,Age}
      * Constraints on nulls
      * Entity integrity constraints
      * Referential integrity constraints
    - Application-based constraints
      * Semantic integrity constraints
        + E.g. salary of an employee should not exceed the salary of the employees supervisor.
      * Cannot be directly expressed in the schemas of the data model
      * Must be expressed and enforced by application program

**DAY 8:**

**Key**

* A key is a minimal superkey – a superkey such that removal of any attribute from k results in a set of attributes that is not a superkey
* E.g. {ssn}
* A relation schema may have more than one key, each of the keys is called a candidate key.
* E.g. fig

**Primary Key**

* Designate one of the candidate keys as the primary key

**Entity integrity constraint**

* Entity integrity constraint: no primary key calue can be null
* Because the primary foregin key key value is used to identify individual tuples in a relation
* Involve a single relation

Regerential integrity constraints

Foreign key

* A set of attributes of fk in the relation schema R1 is a foreign key of R! that references relation R2 if it satisfies two rules:
  + 1. The attributes in FK have the same domain(s) as the primary key attributes PK of R2
  + 2. A value of f kina tuple t1 of the current state r1(R1) either occurs as a value of PK or some tuple t2 in the current state r2(R2), or is null.
  + E.g
    - Employee (ssn… dno) foreign key
    - Department(dnumber, dname,…) primary key

**Relational database**

* Schema== relations + constraints
* State
* Schema cont.
  + S is a set of relational schemas S={r1,r2…,rm}
  + And a set of integrity constraints IC.
* Relational database state DB of S is a set of relation states DB={…?

**Valid /invalid state**

* When we refer to a relational database, we implicitly include its schema and its current state

**Update operations on relations (Cont.)**

* In case of integrity violation, several actions can be taken:
  + Cancel the operation that causes the violation (reject option)
  + Perform the operation but inform the user of the violation
  + Trigger additional updates so the violation is corrected (CASCADE option, SET NULL option)
  + Execute a user specified error-correction routine

**Constraint violation:**

* Insert can violate \_\_\_\_\_\_
  + Domain constraints
  + Key constraints
  + Entity integrity constraints
  + Referential integrity constraints
* Reject the insertion in case of constraint violation
* Delete can violate\_\_\_\_\_
  + Can violate referential integrity
  + In case of violation
    - Reject the deletion
    - Attempt to cascade the deletion
    - Modify the referencing attribute values the cause the violation
      * Set to null
        + Foreign key is part of the primary key
      * Change to reference another valid tuple
    - Specify it in DDL

**Er relational Mapping**

* Er is a conceptual model

**Strong Entity**

* Create a relation or the entity
* Include all the simple attribute and simple composite attribute of a composite attribute
* **See exercise 3 on paper**

**1:1 Relationship**

* Choose one entity to include the primary key of the other entity as foreign key
* Include simple attributes of the relationship
* It is better to choose an entity type with total participation
* **See exercise 4 on paper**

**1:N Relationship**

* The entity at N side includes the primary key of entity at 1 side as foreign key
* The entity at N side includes simple attributes of the relation
* **See exercise 5 on paper Day 9**

**DAY 10:**

**M:N Relationship**

* Create a ne3w relation for the relationship
* Include the primary keys of the two entities as foreign keys
* The combination of two foreign keys as the primary key of the new relation
* **See exercise 6 day 10 notes**

**Weak Entity**

* The primary key of an weak entity is the combination of the primary key of the owner(s) and the partial key of the weak entity
* **See Exercise 7 Day 10**

**1:N recursive relationship**

* The entity includes the primary key of itself as foreign key to represent the recursion
* **See exercise 8 day 10**

**Multivalued Attributes**

* Create a new relation R for a multivalued attribute A (of Entity E)
* R includes an attribute corresponding to A and the primary key of E as the foreign key
* The combination of A and this foreign key as the primary key of R.
* **See exercise 9 day 10**

**Day 12**

**Natural join**

* A equijoin without superfluous attributes
* Any two join attributes have the same name in b othe relations
* Join attributes
* Equating all attributes pairs that have the same name in the two relations
* Rename when necessary before applying nature join
* E.g Dept\_locs=department\*dept locations
* **SEE exercise 1 day 12**

**Day 15**

Examples relational algebra

* **See Day 15 examples**

**DAY 16**

* SQL-86 (SQL 1)
* SQL 92 (SQL 2)
* SWL-99 (SQL 3)
  + Core supposed to be implemented by all rdbms vendors
  + Extension:optional modules such as data mining, spatial data, temporal data

CREATE TABLE:

* Specifies a new base relation by giving it a name, and specifying each of its attributes and their data types
* A constraint not null may be specified on an attribute
* Create table department ( DNAME VARCHAR(10) not null,

DNUMBER INTEGER not null,

MGrSSN CHAR(9),

MGRStartDate CHAR(9) );

Attribute data tupes and domains in sql

* Numeric
  + Integer or int
  + Float or real
  + Decimal (I,j), or Dec(I,j) or numberic(I,j)
    - I: total number of decimal something

Create Table Department (

DNAME Varchar(10) not null, Dnumber integer not null check (dnumber>0 and dnumber<21),

Mgrssn char(9), mgrstartdate Date, Primary key (dnumber), Unique (dname), Foreign Key (mgrssn) references employee (ssn);

**Day 17**

Referential integrity options

* We can specify cascade, set null or set default on referential integrity constraints
* Create talbe employee( …. DNO int not null default 1, priomary key (ssn),

foreign key (superssn) on update set cascade on delete set null,

**Day 18**

Simple Sql Queries (cont)

* Query 1: retrieve the name and address of all employees who work for the ‘research’ department.
* **Example see NOTES DAY 18**
* Query 2: for every project located in ‘staffor’ , list the project number , the controlling department number, and the department managers last name, and birthdate