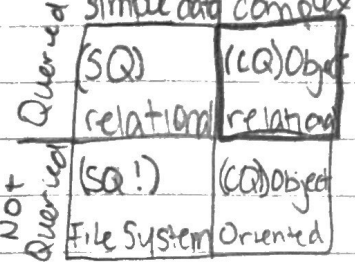
Database Systems (8/22/2023)

Database Requirements

- Collection
- Universe of discourse
- Logically coherent
- Specific purpose
- Source(s)
- Interaction with events
- Actively interested audience
- Generated and maintained manually or by machine
- Any size and complexity

DBMS (Database Management System)

- collection of general purpose software programs
- 2 functions (define the database, manipulate data)



Simple data

- atomic
- no manipulation

Ex: number, word

Complex data

- resulting from calc
- computed

Queried systems

- returns dynamic results

Ex: list of records for data range

Not-Queried Systems

- Provids static view of data recor

Ex: Text files that aren't searchable

When not to build database

- costs increase for same result
- time to execute processes increases
- lacking control & restriction
- difficult concepts & understanding

Database Systems (Ch. 1)

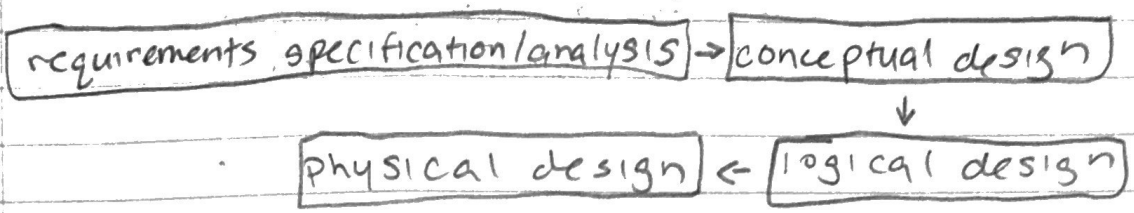
miniworld/universe of discourse - some aspect of real world

meta data - data in DBMS in form of catalog or dictionary

query - causes some data to be retrieved

transaction - data read/written into database

* data records are storing same type, data elements are the categories, data type is string, int, etc



NOSQL systems - contain self describing data with names/values in one structure

operation (function/method) contains interface & method

* users have different view of database; a subset of database or containing virtual data not explicitly stored

concurrency control - ensures the result of several users updating same data is updated correctly

online transaction processing (OLTP)

transaction - process that includes one or more database accesses

isolation - allows each transaction to execute in isolation of other transactions

* administrators, designers, end users, analysts, programmers,

redundancy controlling - same data several times wastes space & is inconsistent

* storing in one place is data normalization, putting a bunch of data into one file is denormalization

* most DBMS do their own data buffering due to

Database Systems (8/24/2025)

3 categories of Data Models
- High level / conceptual model
- Low level / physical model
- Representational / implementation model

schema construct - formal notation to describe entity w/ attributes

ENTITY = OBJECT = TABLE Attribute = Property = Column = Field

INSTANCE = Constructs + Domain = STATE

* changes when add, delete, modify

Database Systems (Ch. 2)

Entity - real world object/concept described through attributes

access path - search structure for efficiency

description of database is database schema, displayed schema is schema diagram

object in schema is schema construct

data definition language (DDL) & storage definition language (SDL) & data manipulation language (DML)

3 Schema architecture

* interfaces differ when provided by DBMS

database & DBMS catalog are stored on disk, accessed by OS, typically with buffer management

* page 43 (PDF 74)

database utilities: loading, backup, database storage recognition, performance monitoring

* accessing from remote location, connection communication networks

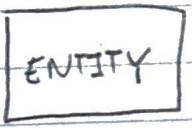
* two tier vs three tier architectures

- three tier adds an intermediate layer between client and database server (application or web server)

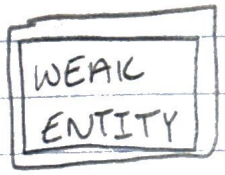
* classification of DBMS

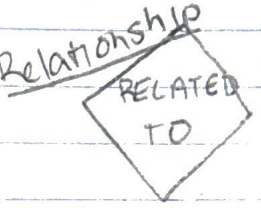
Database Systems (8/29/2023)

ER diagrams are a formal way of representing:
- objects
- properties
- relationships
- constraints



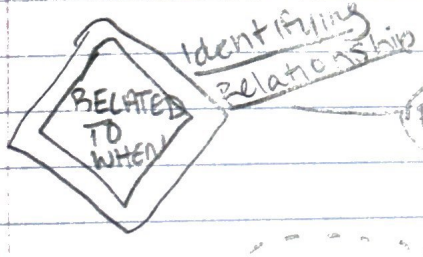
- rectangle with a name (usually a noun)



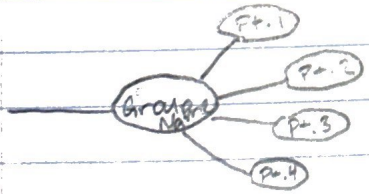


- connects two entities (often a verb)

* never enter a baby without a mother



Derived - complex attribute data.



Composite attribute - several atomic parts must be collected together

The Role —— connects entity with relationship

The mandatory role ═══ MUST participate in relationship

(min, max) constraint (min, max) - tells what the min & max participation in a relationship might be

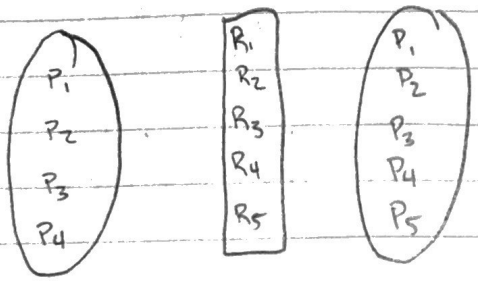
The cardinality constraint (C) - shows ## of members in entity that can be related to members in another entity

Database Systems (8/29/2023)

Key — Key attribute - property that is always unique in value

F-Key — Foreign Key attribute - not necessarily unique, but used as matching value

* order of steps for constructing diagram



Database Systems (Ch. 3)

entity - thing or object

attributes - particular properties they describe

* EMPLOYEE entity might describe name, age, address, salary, job

simple (atomic) attributes - attributes no longer divisible

single-valued - attributes have single value for entity

derived attribute - derivable from stored attribute (you can get age from birthdate)

* complex attributes
* relationships (binary & ternary, recursive relationships, cardinality ratio
* UML class diagrams
* ER model on page 85, 89, 91, 94, 98, 99, 102

Database Systems (Ch. 16)

hierarchy:
* Primary storage, secondary storage, tertiary storage
  ↓ cache memory, DRAM, magnetic disks, CD-ROM, DVD, tapes
  Flash memory, optical drives, magnetic tapes
* secondary storage devices are nonvolatile, main memory is volatile

Data access more efficient on disk: buffering, organization, reading ahead of request, scheduling I/O, log disks, recovery purposes

pointer and then spanned

* accessing disk block is expensive because of seek time, rotational delay, block transfer time
* double buffering when accessing continuous disk blocks
* fixed / variable length, spanned / unspanned, file header
* 3 file organizations: ordered, unordered, hashed

Database Systems (9/5/2023)

Disks - works with read/write head to store/retrieve data
Tracks - concentric circles on disk
Cylinder - tracks with same diameter on consecutive disks
Blocks - range from 512 bytes to 4096 bytes; equal divisio. of a track set by the operating system

Solid State Drives

- use electronic circuits to store/retrieve data; store data in blocks; no mechanical components; embedded processo
- SSDs read & write data to an underlying set of interconnecte flash memory chips; store data even when it's not connecte to a power source
- SSD's are able to read blocks of data that are spread out

Seek time, Block transfer time, Latency, Bulk Transfer Rate

* Total Time = Seek + Latency + Block transfer time
  Large ~ 10 msec Large ~ 10 msec Small ~ 1 msec

Variable Length record - unknown size vs Fixed Length

Mixed File - variable length field, multivalued attribute, optional

bfr = floor(B/R) B = block size in bytes R = record size in byte
UNUSED = B - (bfr * R)

* red - record at a time blue - set at a time

Database Systems (9/7/2023)

1. Look at the parent
2. Determine if it has an attribute
   a) Yes? What domain needs to be true
   b) No? Add it, indicate domain
3. make subclasses
4. Connect role in to circle.

...

* 2 reasons to make subclasses (collect special properties, participate in a special relationship)

Database Systems (Ch. 4)

superclass/subclass or supertype/subtype or class/subclass

type inheritance - subclass inherits from class

generalization - making a generalized entity from what's given (seeing truck and car then making vehicle class)

predicate defined, attribute defined, user defined

disjointness constraint - entity can be a member of 1 subclass(d)

overlapping constraint - entity can be member of >1 subclass(o)

partial specialization - entity belongs to no subclasses

specialization lattice - subclass in >1 class/subclass relationship

top-down vs bottom up conceptual

*union

superclass/subclass (IS-A) relationship $S \subseteq C$

identification - classes/objects made known by an identifier

Specialization - classifying a class of objects into more specialized subclasses

Generalization - generalizing classes into a higher class

aggregation & association

ontologies & semantic web

Database Systems (Test 1)

Ch 1:

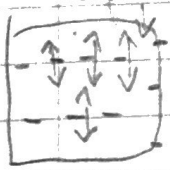
* ACID (atomicity, consistency, isolation, durability)
* DBMS has 2 functions: define database, manipulate data

Database + DBMS = Database System

	Simple	Complex
Queried	Relational	Object relational
Not Queried	File System	object oriented

Ch 2:

* High level/conceptual model: shows relationships, how the user perceives the database (ER, EER)
* Low level/physical model: shows how data is stored, (network diagrams, paths, etc.)
* Representational model: shows how it will be set up to work (only model with view-output) (3 schema)



External schema: shows views-output
Conceptual schema: formal schema constructs (entities, attr...)
Internal schema: physical place for database

* DBMS language sets: DDL, SDL, VDL, DML
* User interface: DBA, GUI, MENU BASED, FORMS BASED

Ch 3:

* ER diagrams are formal way of representing objects, properties, relationships, constraints
- Look through all symbols (names, minimax, C, assumptions

Ch 16:

* Primary storage: operated by CPU, fast access, limited storage
* Secondary storage: not processed by CPU, slower access, much larger storage capacity
* Solid state drives (SSD): electronic circuits to store/retrieve data, store data in blocks, no mechanical components, embedded processor, read/write to set of inter... of flash chips, store data when not connected to power
* Seek time: time to position read/write heads or to a...
* Block transfer time: time to load block from secondar...

to primary storage
* Latency: time it takes for block to rotate into position under read/write head
* Bulk transfer rate: time to transfer consecutive blocks
* TOTAL TIME = SEEK + LATENCY + BLOCK TRANSFER TIME
large ~10 msec large ~10 msec small ~1 msec
* Bits → Bytes → Blocks → Records → Files → Schema (directory)
* voice (2,000), date time (18 bytes)
* variable length (unknown size - voice has 0-2,000)
* fixed length never changes size
* variable records happen when variable length field, multivalued attribute, field is optional, mixed file
* blockfactor = floor(block size / record size)
* unused = block size - (blockfactor * record size)
* spanned vs unspanned blocks
* b = ceiling (r / bfr)
* Hashing: distribute records uniformly, minimize collisions, not leave unused space
* operations on files: retrieve/read/expand, file header carries information, searching begins at file header
* Simple selection (only new voicemails) vs complex selection (includes several simple conditions)
* file organization: organize data into records, blocks, ptrs
* access methods: group of programs that allow operations on data

Ch 4 * EER model advantages: inheritance, minimize redundancy
implementation order
* open end ∪ points to parent
* defining predicate (# of legs)
* boolean condition of predicate (# of legs = 0)
* d is disjoint, o is overlapping
* 2 reasons to make subclasses: to get special properties, participate in a special relationship

* How to determine bottom up vs top down; what does source refer to # of assumptions

* Formal way : <PARENT>/<SUBSET>
* Unions: every thing from all sets
* intersections: distinct items common to all
* superclass ; parent
* specialization- top down
* generalization - bottom up

Database Systems (9/19/2023)

Primary indexes

* primary key, <value, ptr> pairs stored in blocks, one entry for every block, total index entries = total # of disk blocks, block anchor (first record in block), non-dense (not an index entry for every record)
* Advantage: smaller, fewer block accesses
* Disadvantage: insertion/deletion may cause records to be redistributed

Secondary indexes:

* non-ordering attribute, one distinct index (candidate key), not distinct (build more than one index on same attribute), <value, ptr> pairs in blocks, dense indexes (one entry for every record), total index entries = total # of records
* Advantage: improves search for arbitrary record
* Disadvantage: index is dense so it requires more space and more time for searching

Clustering indexes:

* common ordering attribute (foreign key), <value, ptr>, non-dense index (one entry for every distinct value), total index entries = total distinct values
* Advantage: search only records that hold true for index value
* Disadvantage: insertion/deletion are expensive

Logical indexing:

* secondary index

Physical indexing:

* Primary / Clustering indexes

B-Tree

* B for balanced, node & sub-node, levels (0-n)
* Highest level = leaves
* internal nodes - leaves (lower than leaf level)
* level 0 is root node

Database Systems (9/28/2023)

Relational model - words, schema constructs, symbols, mathematical statements

Terminology - relation, tuple, attribute, domain, current relational state, roles, NULLs, degree

NULL - unknown but does exist, not applicable in record, doesn't exist and cannot ever be filled

* New constraints become important: no multivalue or composite attributes

* sets have no order

R - name of relation Q, R, S - denote different relation names

r(A) - the relationship known as R q r s - denotes state of entities Q, R

t[An] - value of the n^{th} attribute in row

R.A - identifies relation with its attribute t u v - used to denote tuples of different types

dom(R) = X DOMAIN CONSTRAINTS - specify an allowable range of values

$t_1[S.K_1] \neq t_2[S.K_2]$ KEYS - define relational constraints (elements of set are distinct)

SUPERKEY - subset of all attributes for a relation

MINIMAL SUPERKEY - smallest set of attributes

CANDIDATE KEY - deploy an index to work through records quickly

Relational database schema = set of schemas + set of integrity constraints

* Integrity of a database
↳ methods to promote better integrity: auto-filling, special key structures, disabling user entry, error messaging, match, disallowing deletion of records, forcing deletion

Transaction - some executing program that includes database operations

JOIN - shows things different in relationship
* new relation returned

$r(Q) \leftarrow R \bowtie_{\text{<boolean conditions>}} S$

- $r(Q)$: n+m attributes
- R: n attributes
- $\bowtie_{\text{<boolean conditions>}}$: $R.PK = S.FK$
- S: m attributes

Theta join $\bowtie$ any boolean equalities & inequalities $<, >, \neq, =, \leq, \geq$

Equijoin $\bowtie$ only boolean equalities AND & OR

Natural join * only boolean equalities, duplicated values left out AND & OR

Database Systems (10/3/2023)

left outer join

$Q \leftarrow R \bowtie_{R_1 = S_1} S$

Returned:

- every raw from R
- matching raws from S and NULL values where S matched tuple in R

right outer join

$Q \leftarrow R \bowtie_{R_2 = S_3} S$

Returned:

- every match from R, all other rows from S & NULL values where nothing from R matched tuple in S

full outer join

Returned:

- Every row from R with & without matches
- Every row from S

union

- combining two sets (no duplicates)

intersection

- rows that appear in both

difference

- keep the rows in R that aren't in S