Relational Model.

Book 5.1 8.9.

entity Integrity const: no Primary key can be NULL

referential integrity Const: a tuple in relation A referring to relation B must refer to an existing tuple in B.

1. Inherent model — no duplicates tuples
   - Data structures
   - Constraints
   - Operation
2. schema based, explicit — superkey, unique key, entity integrity, referential...
   - Algebra
3. application based / semantic
   business rules

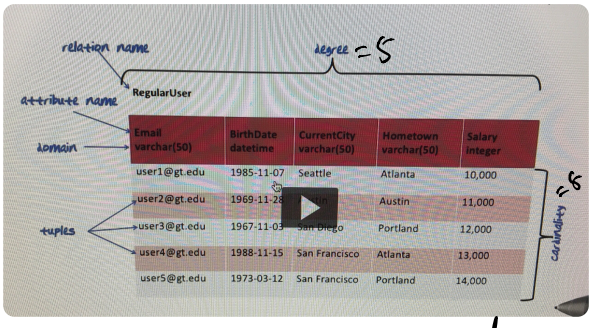
Calculus
- tuple calculus — sql
- domain calculus (QBE)

Data structure:

a domain D: a set of atomic values (value has no meaning)

a relation R: a subset of set of ordered n-tuples (a column of values)
$R \subseteq (d_1, d_2 \ldots d_n \mid d_i \in D_i$, i in ...)

an attribute A: a unique name given to a domain in a relation. helping us interpret domain values (a table; column header)
It is possible for several attributes have same domain.



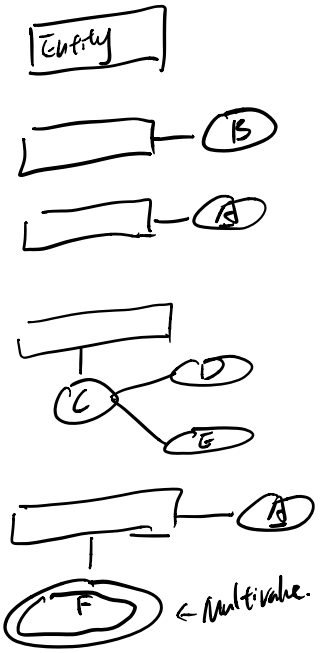
de

* The value of a relation is independent of attribute order & tuple order

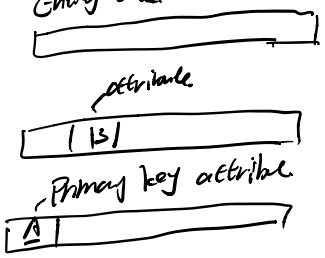
EER Relational Mapping.

From EER model to Relational table.

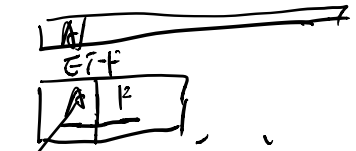
EER notation



Relational mapping

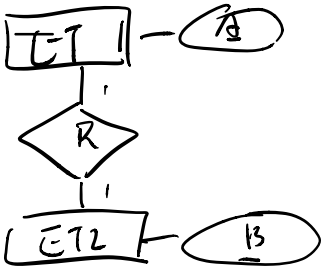


(Property type get lost)

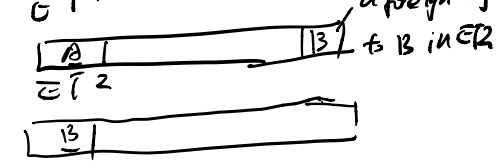


A & F is a composite key of ET-1

Attribute A in ET-1 table is a foreign key to attribute A in ET table



method 1:

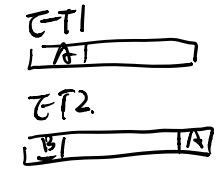


or method 2:

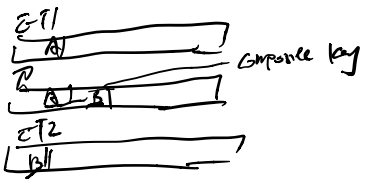
∵ mandatory participation of ET2, we can only use method 2.

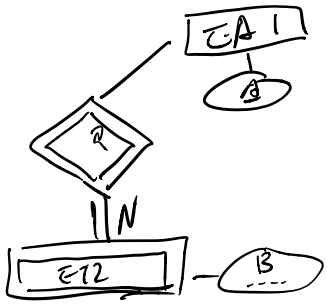
because. method 2 will show all B to A

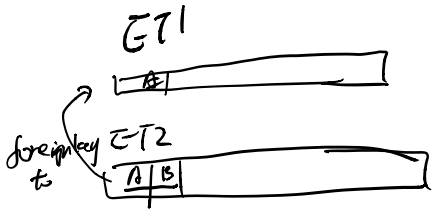
ET1 <-> ET2
1 to N

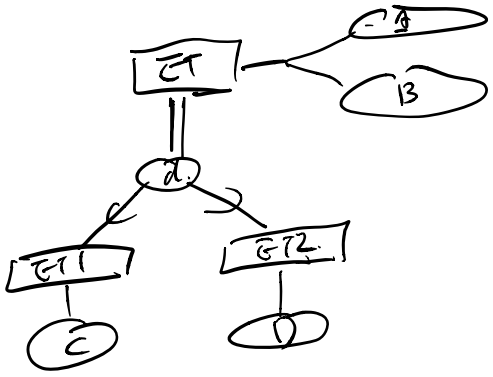


N to M.

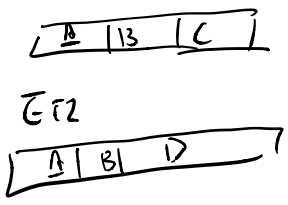


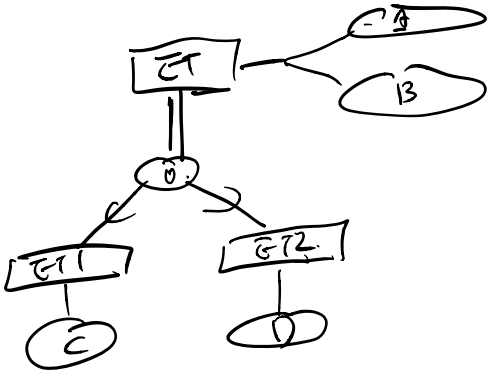




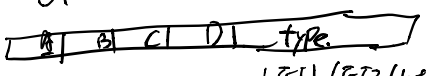


C can omit B and add a ET table)





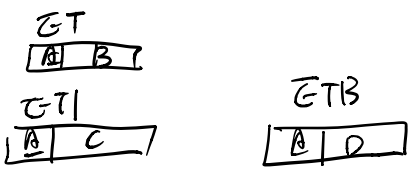
Options (don't like)



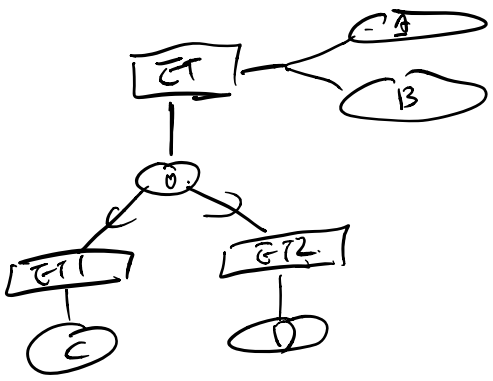
(ET1 / ET2 / both)

- Lots of null.
- Consistency problem of type. column

or



n-n mandatory



Option 1:
Single table

A	B	C	D	type.

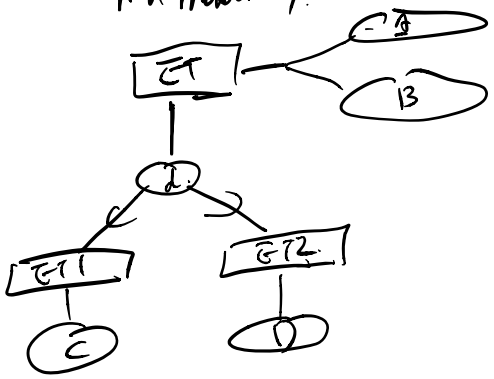
Or

ET1	
A	B

ET1	
A	C

ET2	
A	D

n-n mandatory

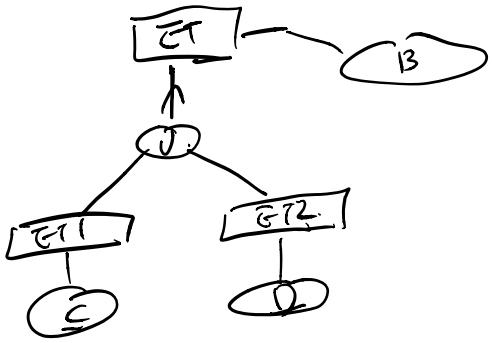


ET1	
A	B

ET1	
A	C

ET2	
A	D

Union type



ET

ET-ID	B

Correspond to either C/D

ET1	
C	ET-ID

ET2	
D	ET-ID

Relational Algebra & Calculus

Relational Algebra Ops

$R \cup S$ union

$R \cap S$ intersection

$R \setminus S$ set difference

$R \times S$ cartesian Product

$\rho_{[A_1, B_1 \ldots A_n B_n]}$ rename.

$\pi_{A_1, A_2 \ldots A_n}(R)$ projection

$\sigma_{expression}(R)$ selection

$R * S$ natural join

$R \overline{\bowtie} S$ outer join(s)

$R \bowtie_\theta S$ theta join

$R \div S$ divided by

Selection: σ

simple expressions: $= \quad < \quad > \quad \leq \quad \geq \quad \neq$

composite expressions: expression AND expression2 $\wedge$

OR $\vee$

(expression)

Not expression

Projection - $\pi_{A_1, A_2 \ldots A_n}(R)$ No Duplicates

eg Find email, birthyear, & sex for regular users in Atlanta.

$\pi_{Email, BirthYear, Sex}(\sigma_{HomeTown = \text{'Atlanta'}}(\text{RegularUser}))$

Relations are Sets!!!

eg Find Sex for regularUsers in Atlanta., we only have 2 answers, male or female

Union - $\cup$ (or) same degree & same domain for each

$\pi_{CurrentCity}(\text{RegularUser}) \cup \pi_{HomeTown}(\text{RegularUser})$

must be type compatible. i.e. same number of attributes && pairwise same types

$\cap$. (and)

Set difference - $\setminus$

$A \setminus B \rightarrow$ 

natural Join $*$ (inner join)

- matches attri with same name.
- keep only 1 copy of common attributes

Theta join - $\bowtie_\theta$

e.g. Regular User $\bowtie$ Birthyear < Grant for Major 60s Grants

(table 1 — expression — table 2)

combine records with expression satisfied records

- both attribute appears
- inner join

Outer-Join:

left: $\overrightarrow{\bowtie}$

preserve left records even if not in right.

Cartesian Product - $\times$ (natural join w/ no same name columns)

each record in table 1 combine with those in table 2.

(useful to find complement set)

Divide by - $\div$

$R(A,B) \div S(B) = \{r.A \mid r \in R \text{ and } \forall (s \in S)\, \exists (t \in R)$

$(t.A = r.A \text{ and } t.B = s.B)\}$

(table_attr: R(A,B); table_attr: S(B))

e.g. R A B user | interest

$\pi_{Name, Interest}(UserInterest) \div \pi^{S}_{Interest^{B}}(Email = \text{'user1@gc.edu'}(UserInterest))$

meaning: Find Gmail of all users w/ at least all the interest of user 1

Rename - ρ

ρ RUser [Year Birthday, Gender sex] (Regular User)

tuple rename.

Calculus:

Relational calculus Expressions

$\{t \mid P(t)\}$

range expression: $t \in R$ and $R(t)$ denotes t is a tuple of relation R

attribute use: $t.A$ value of t on attribute A

constant: c.

comparison operators Θ: $=, \neq, \leq, \geq, <, >$

atoms: $t \in R$, $r.A \Theta s.B$, or $r.A \Theta c$

Predicate: an atom is a predicate; if P_1 & P_2 are predicates, so are (P_1), NOT(P_1), P_1 or P_2, P_1 and P_2, $P_1 \Rightarrow P_2$

Selection in tuple calculus:

$\{r \mid r \in \text{Regular User}\}$

$\{r \mid r \in \text{Regular User}$ and $(r.\text{CurrentCity} = r.\text{HomeTown}$ or $r.\text{HomeTown} = \text{'Atlanta'})\}$

Projection:

$\{r.\text{Email}, r.\text{BirthYear}, r.\text{Sex} \mid r \in \text{Regular User}\}$

Union:

$\{s.\text{City} \mid \exists (r \in \text{Regular Users})(s.\text{City} = r.\text{CurrentCity})$ or $\exists (t \in \text{Regular Users})(s.\text{City} = t.\text{HomeTown})\}$

Intersection.

Diff: $\{ s.city \mid \exists (r \in RegularUsers)(s.city = r.current\_city)$
and Not$(\exists (t \in RegularUsers)(s.city = t.HometownCity))\}$

Natural Join:

$\{ t.Email, t.Year, t.Sex, t.Event \mid \exists (r \in RegularUser) \exists (s \in MajorGoEvents)$ $(r.Year = s.Year$ and $t.Email = r.Email$
$t.Year = \ldots$ and $t.Sex = r.Sex$ and $t.Event = s.event)\}$

join expression

get columns from RegularUser

get columns from MajorGoEvents

Cartesian Product.

$\{ r, s \mid r \in RegularUser$ and $s \in UserTweet \}$

Divide by

SQL. ✱ SQL can Have duplicate Rows!!!

Insert:

insert into table (c_1, c_2, c_3) VALUES (v_1, v_2, v_3)

Delete:

Delete From Table. (where ---)

can delete a set of rows

Update:

Update Table. SET $c_1 = v_1$ where $c_2 = v_2$ AND $c_3 = v_3$

General syntax

SELECT $c_1, c_2 \ldots$ From $t_1, t_2 \ldots t_n$ Where condition

Distinct:

Select Distinct (c_1) from t_1

Natural Inner Join

Need to specify columns from which table if it exists in both tables.

With NATURAL JOIN clause, you done need specify joined column name

LEFT Outter Join

SELECT c_1 , $t_1.c_2$, c_3. From t_1 left Outer Join t_2

String Matching

LIKE 'San%', % matches any string, including empty string.

LIKE 'A_____' _ matches single char

Sorting.

ORDER BY ASC/DESC.

Union.

```
SELECT c1 From t1
Union
SELECT c2 From t2
```

(No duplicates)

(Duplicates Union All)

Intersect

```
SELECT c1 From t1
Intersect
SELECT c2 From t2
```

Intersect All will include duplicates

Except

```
SELECT c1 From t1
EXCEPT
SELECT c2 From t2
```

see diff.

Except All will include duplicates

Having: put condition on group. (groupby)

Nested Queries:

In / Not In

Where C_1 > All (another Query)

or > (select max( ))

Correlated

```
select R.Email, Birthday
    From Regular User R
    where Not Exist
    (select * From UserInterest U where U.Email = R.Email)
```

reference out of scope. ...

When R email ∈ U email.

For each row, see whether there is empty set