Date 11/11/2024
Page 11

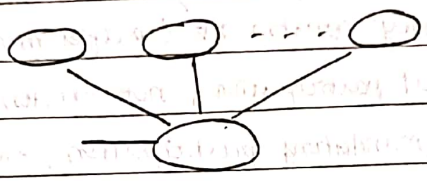
Unit II

* 2 main activities :
  i) Database design
  ii) Application design

9/11/24

Notation for ER diagrams :

1) Entity :
2) Weak entity : ex : DEPENDENT
3) Relationship :
4) Identifying relationship : Relationship between a strong and weak entity.
5) Attribute :
6) Key Attribute :
7) Multivalued attribute : Ex ; email , phone no
8) Composite attribute :



Ex : Name , address
9) Derived attribute : Ex ; age (derived using DOB)

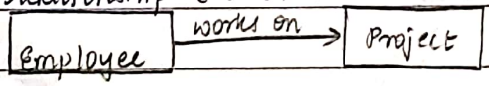
ER model has 3 main concepts :
1) Entities (types and entity set)
2) Attributes (simple , composite , multivalued)
3) Relationships (type and relationship set)

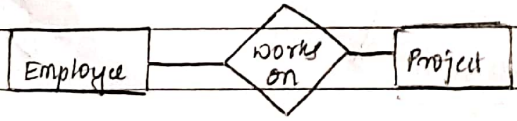
Relationship types :

Degree of relationship : no of entities participating

1) Binary relationship (2 entities are involved)

ex :





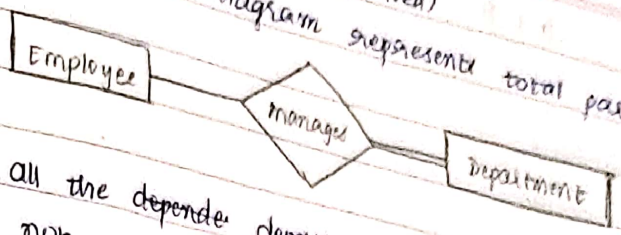
Scanned with OKEN Scanner

should have total participation.

Ternary relationship (3 entities are involved)

Note: Double line in ER diagram represents total participation

Example:



Here, all the dependent departments should have a manager but all employees are not managers.

Constraints on relationship (Ratio constraints)

(i) Cardinality ratio: (specifies maximum participation)

(a) One-to-one (1:1)

(b) One-to-many (1:N) or Many-to-one (N:1)

(c) Many-to-many (M:N)

(ii) Existence dependency constraint (specifies minimum participation) → (OR) Participation constraint

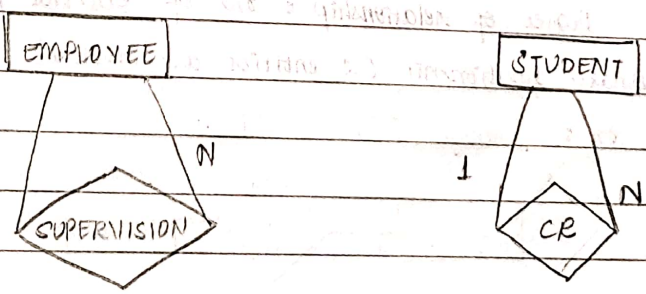
(a) zero (optional participation, not existence-dependent)

(b) one or more (mandatory participation, existence-dependent)

Note: Relationship between the different instances of same entity is called recursive relationship.

Recursive relationship type:

* A relationship type btw the same participating entity type in 'distinct roles'
* Also called a 'self-referencing' relationship type.
* example: the SUPERVISION relationship. ;, CR relationship.

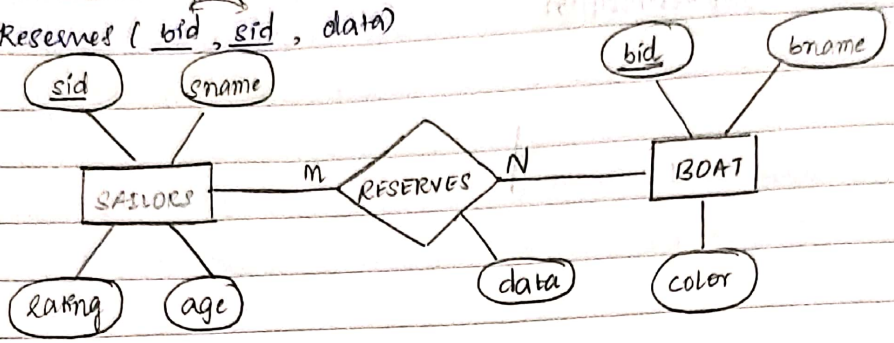


Scanned with OKEN Scanner

* An entity that does not have a key attribute and that is identification dependent on another entity type is called weak entity.
Entities are identified by the combination of :
* A partial key of the weak entity type
* The partial entity they are related in the identifying the relationship particular type.

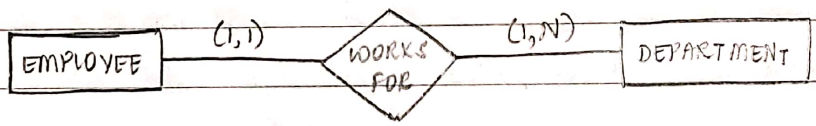
If the cardinality ratio is M:N, the relationship should be created as table.

Sailors database
Sailors (sid, sname, rating, age)
Boat (bid, bname, color)
Reserves (bid, sid, data)



create table sailors (sid int primary key, sname varchar(20), rating int, age int);
create table boat (bid int primary key, bname varchar(20), color varchar(10))
create table reserves (sid int references sid of sailors, bid int refernces bid of boat, date varchar(10), primary key sid, bid);

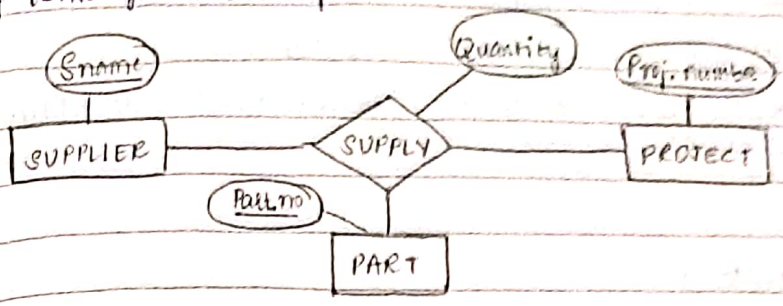
13/11/24 The (min, max) notation for relationship constraints :



Scanned with OKEN Scanner

Relationship of degree - n are called n-ary.

Example of ternary relationship :



Constraints on higher-degree relationship in general has both (min, max) and 1 or M or N representation.

Relational model concepts.

Relational model:

The relational model of data is based on the concept of a relation.

* Given $R(A_1, A_2, \ldots, A_n)$ — relation → set of attributes; dom → set of valid values

$r(R) \subset dom(A_1) \times dom(A_2) \times \ldots \times dom(A_n)$ ← instance or tuple

(i) $R(A_1, A_2, A_3, \ldots, A_n)$ is the schema of the relation.

Example:

Let $R(A_1, A_2)$ be a relation schema:

Let $dom(A_1) = \{0, 1\}$

Let $dom(A_2) = \{a, b, c\}$

Then: $dom(A_1) \times dom(A_2)$ is all possible combinations:

$\{<0,a>, <0,b>, <0,c>, <1,a>, <1,b>, <1,c>\}$

* Table definition is called 'schema of a relation'.
Populated table is called 'state of a relation'.

16/11/2024 Characteristics of relation:

1. Ordering of tuples in a relation $r(R)$:
The tuples are not considered to be ordered.

2. Ordering of attributes in a relation schema R:
We'll consider the attributes in $R(A_1, A_2, \ldots, A_n)$ and the values in $t = <v_1, v_2, \ldots, v_n>$ to be ordered.

3. Values in a tuple: cannot have composite or multivalued attributes.
* All values are considered atomic (indivisible)
* Each value in a tuple must be from the domain of the attribute for the column
If tuple $t = <v_1, v_2, v_3, \ldots v_n>$ is a tuple(row) in the relation state r of $R(A_1, A_2, \ldots, A_n)$
Then each v_i must be a value from $dom(A_i)$.

Scanned with OKEN Scanner

Date 11/11/24
Page 13

classmate
Date 14/11/24
Page 16

Notation :

We refer to component values of a tuple t by :
t[Ai] or t.Ai
This is the value vi of attribute Ai for tuple t

Constraints :

3 main types of constraints :
(i) Inherit or Implicit constraints
(ii) Schema-based or Explicit constraints
(iii) Application based or Semantic constraints

There are 3 main types of (explicit schema-based) constraints that can be expressed in the relational model :
* Key constraints
* Entity integrity constraints
* Referential integrity constraints.

Key constraints :

1. Superkey of R :
Is a set of attributes Sk (superkey) of R with the following condition :
* No 2 tuples in any valid relation state r(R) will have the same value for Sk.
* That is, for any distinct tuples t1 and t2 in r(R), $t1[SK] \neq t2[SK]$
* This condition must hold in any valid state r(R).

2. Key of R :
* A 'minimal' superkey.
* A key is a superkey K such that removal of any attribute from K results in a set of attributes that is not a superkey.
* A key is a Superkey but not vice versa.

Note : Any set of attributes that includes a key is a superkey.

Scanned with OKEN Scanner

$E \rightarrow 5+T|3-T$

$T \rightarrow V$

$\dots\rightarrow VT'$

After Rewrite -

foreign key (SID) references student(SID) on delete cascade
on delete set-null
on update cascade
on update set-null

diagemate
Date 19/11/24
Page 17

* If a relation has several candidate keys, one is chosen arbitrarily to be the primary key.

Entity integrity constraints:
* The primary key attributes of each relation schema R in S cannot have null values in any tuple of r(R).
* If primary key has several attributes, null is not allowed in any of these attributes.

Referential integrity constraints: (Foreign key constraint)
* A constraint involving 2 relations.
* Tuples in the referencing relation R_1 have attributes foreign key that reference the primary key attributes of the referenced relation R_2.
→ A tuple t_1 in R_1 is said to reference a tuple t_2 in R_2 if
$t_1[FK] = t_2[PK]$.
* Domain for foreign key should be equal to the values in the referring primary key.
* The value of in the foreign key column foreign key of the the ~~referring~~ referencing relation R_1 can be either:
(i) a value of an existing primary key value of a corresponding primary key PK in the referenced relation R_2 (or)
(ii) a null

20/11/24

Other constraints:
1. Semantic integrity constraints
* semantic constraints cannot be expressed on ER diagram.

Update operations on Relations:
* Integrity constraints should not be violated by the update operations.
* Several update operations may have to be grouped together.
* Updates may propagate (cascade) to cause other updates automatically. This may be necessary to maintain integrity constraints.

Scanned with OKEN Scanner

In case of integrity violation, several actions can be taken:
(i) cancel the operation that causes the violation (RESTRICT or REJECT option)
(ii) perform the operation but inform the user of the violation.
(iii) trigger additional updates so the violation is corrected (CASCADE option, SET NULL option) - for one table maximum of 6 triggers can be created.
(iv) execute a user-specified error correction routine.

Possible violations for each operation:
Insert may violate any of the constraints:
(i) Domain constraint.
(ii) key constraint
(iii) Referential integrity constraint - occurs in referencing table.
(iv) Entity integrity - if primary key of newly inserted tuple is zero.
Delete may violate only referential integrity
Remedy: RESTRICT, CASCADE, SET NULL
Update may violate domain constraint and NOT NULL constraint on an attribute being modified.
Any of the other constraints may also be violated, depending on the attribute being updated:
(i) Updating the primary key
(ii) Updating ordinary attribute
(iii) Updating the foreign key.

classmate
Date 21/11/2024
Page 19

E-R to relational mapping algorithms :

Step 1 : Mapping of regular entity types (strong entity)

Step 2 : Mapping of weak entity types

Step 3 : Mapping of Binary 1:1 relation types

Step 4 : Mapping of Binary 1:N relationship types

Step 5 : Mapping of Binary M:N relationship types

Step 6 : Mapping of multivalued attributes

Step 7 : Mapping of N-ary relationship types

Mapping of regular entity types :

* For each regular entity type E in the ER schema, create a relation R that includes all simple attributes of E.

Mapping of weak entity types :

* For each weak entity type W in the ER schema with owner entity type E, create a relation R and include all simple attributes of W as attributes of R.
* Also include as foreign key attribute of R the primary key attribute of the relation that correspond to the owner entity type.
* The primary key of R is the combination of the primary key of the owner and the partial key of the weak type of W, if any.

Mapping of Binary 1:1 relation types :

* For each binary 1:1 relationship type R in the ER schema, identify the relations S and T that correspond to the entity types participating in R. There are 3 possible approaches :

(i) Foreign key approach (2 relations approach)

(ii) Merged relation option (1 relation) : appropriate when both participants are total

(iii) Cross reference or relationship relation (3 relations) option

Mapping of Binary 1:N relationship types :

* For each regular binary 1:N relationship type R, identify the relation S that represent the participating entity type at the N-side of the relationship type.

Scanned with OKEN Scanner

include as foreign key in S the primary key of the relation T that represents the other entity type participating in R.

include any simple attributes of the 1:N relation type as attributes of S.

Binary of M:N relationship types:

for each regular binary M:N relationship type R, create a new relation S to represent R. This is a relationship ~~ship~~ relation.

include as foreign key attributes in S the primary keys of the relations that represent the participating entity types; their combination will form the primary key of S.

Mapping of multivalued attributes:

for each multivalued attribute A, create a new relation R.

This relation R will include an attribute corresponding to A, plus the primary key attribute K - as a foreign key in R - of the relation that represents the entity type or relationship type that has A as an attribute.

Mapping of N-ary relationship types:

for each N-ary relationship type R, where $n > 2$, create a new relationship S to represent R.

include as foreign key of all the primary key attributes of other relations and also include the additional attribute.