Database Systems
The Relational Model
Enhanced Entity-Relationship Model (EER)

Enhanced Entity-Relationship Model (EER)

- EER is also know as database model
- EER is different from ERD because EER defines in detail the physical architecture of the database. ERD, however, is a conceptual design. ERD are never completed.
- EERs are totally based on ERDs.
- EERs define in detail: Primary keys (PK), Foreign Keys (FK), Unique Keys (UN), and all the domain (data type and range) from all the attributes in tables.

- EERs also define the DELETE and UPDATE relationship that two or more entities have in the table.

- EERs drive the SQL code needed to create our physical tables in the system

Enhanced Entity-Relationship Model (EER)

- Keys (strong and weak)
- Strong:
- **Primary Key (PK):** when an attribute in a table is set to PK it cannot be duplicated and must be not NULL by definition. Its main purpose is to identify a record in the table. An entity can have more than one attribute set to PK. Primary keys can be referenced by Foreign Keys
- **Unique Key (UN):** also know as candidate key. When an attribute in a table is set to PK, it cannot be duplicated. However, it can be NULL by definition. Unique keys can be referenced by Foreign keys. Note that UN key is also a weak key when it is set to NULL.

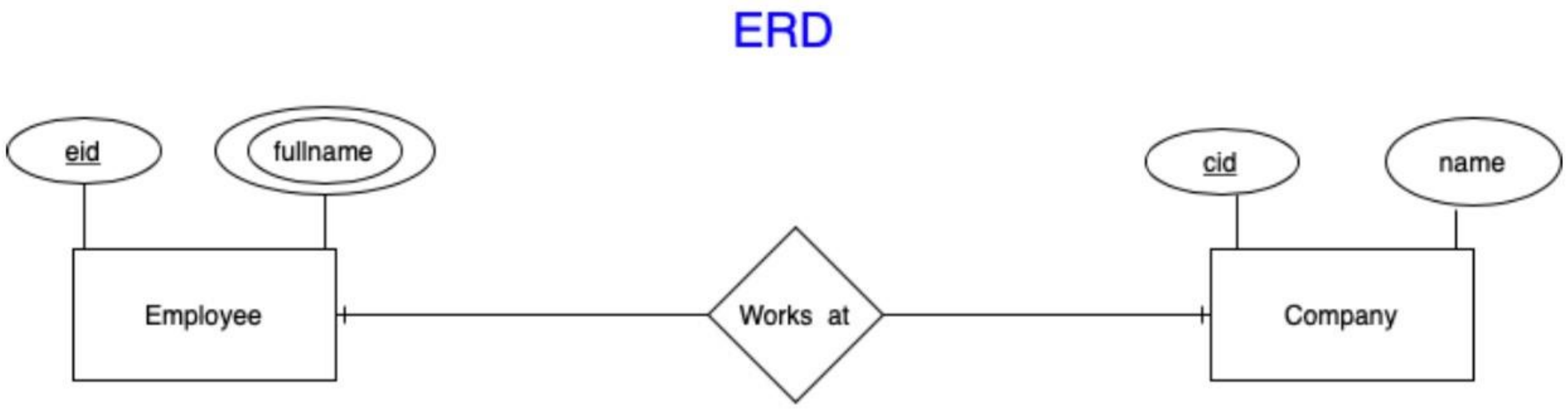
- **Check Key (CK):** also know as check constraint, it limits the range of values for a specific attribute. For example, CHECK (salary > 0). CK is defined at the time the attribute holding the key is created, and is triggered before inserting in the defined attribute.

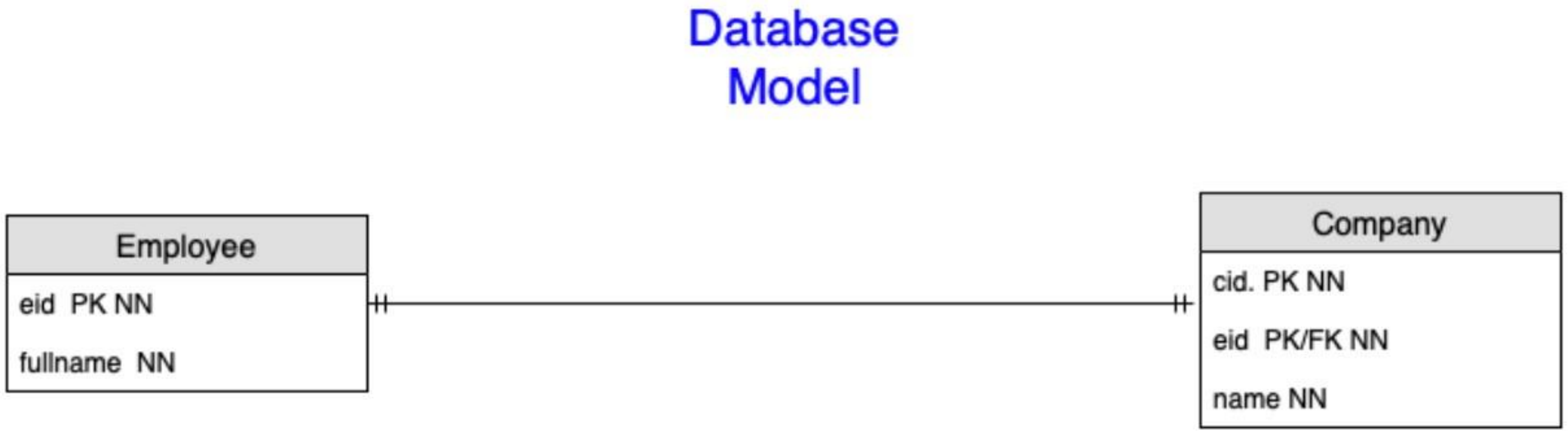
- Weak:

- **Foreign key (FK):** references a PK or UN key from other table (parent table). FK keys <u>can be NULL or can be used together with PK or UN this action is also know as compound key.</u> FK are normally subjected to DELETE AND UPDATE constraints that define the weakness of the key.

FROM ERD TO EER

One to One

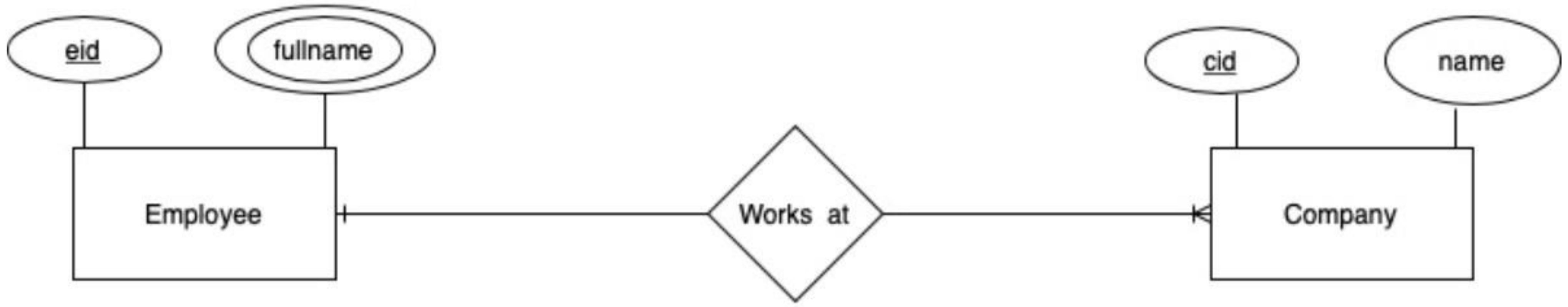




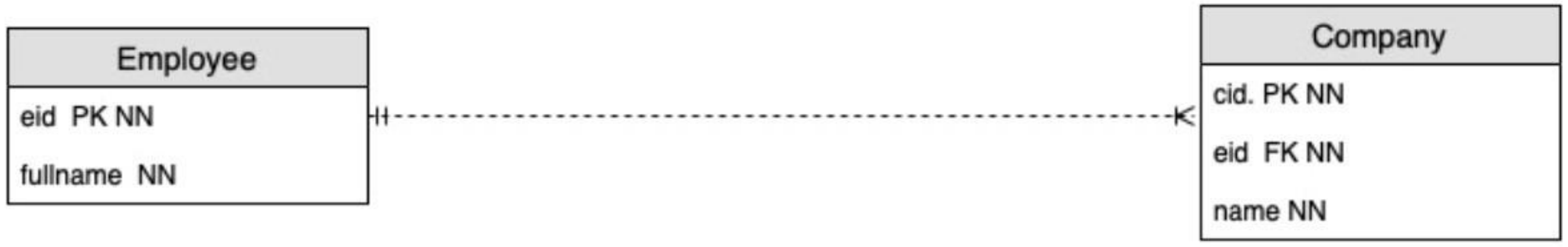
FROM ERD TO EER

One to Many

ERD



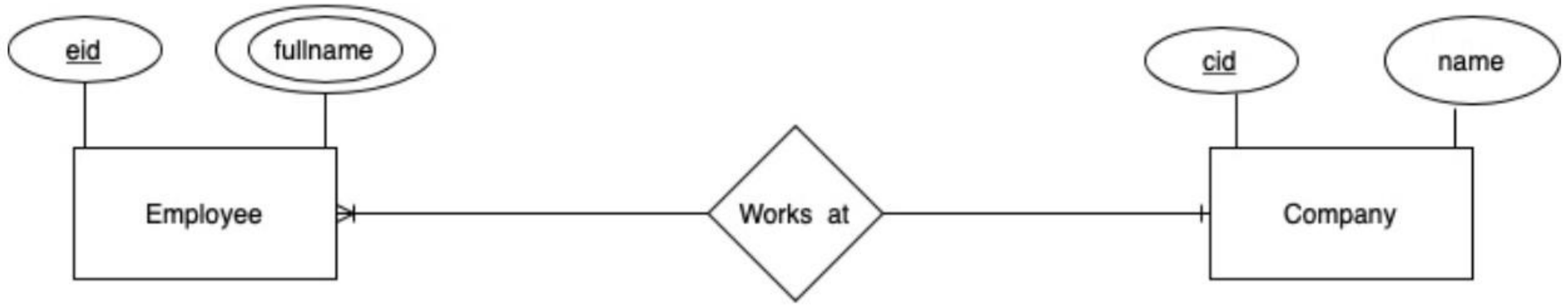
Database Model



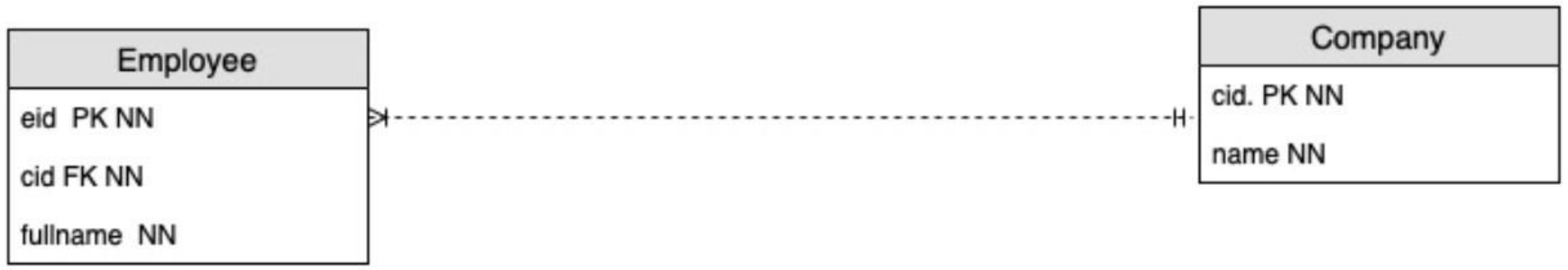
FROM ERD TO EER

Many to One

ERD



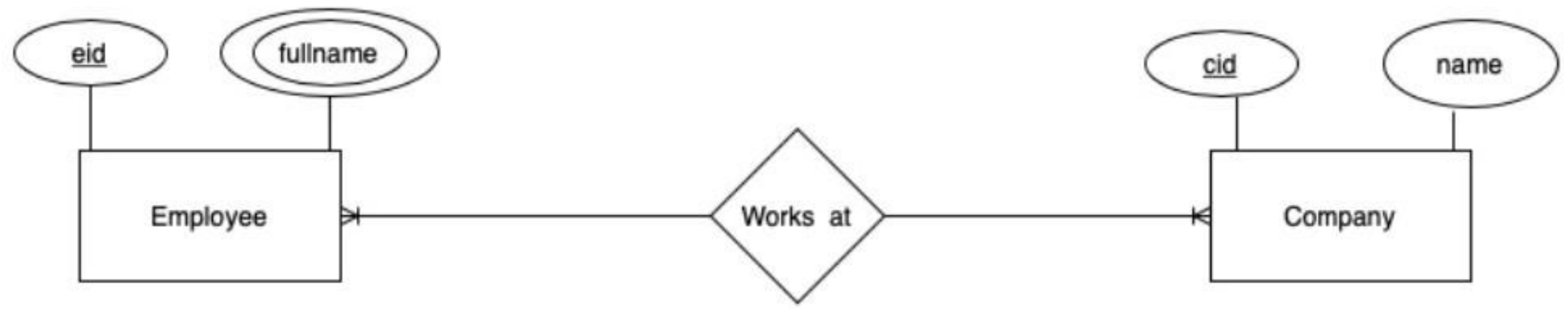
Database Model



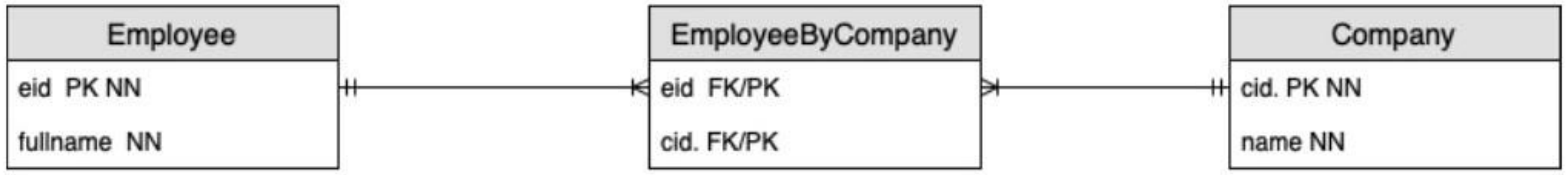
FROM ERD TO EER

Many to Many

ERD



Database
Model



Employee
eid PK NN
fullname NN

EmployeeByCompany
eid FK/PK
cid. FK/PK

Company
cid. PK NN
name NN

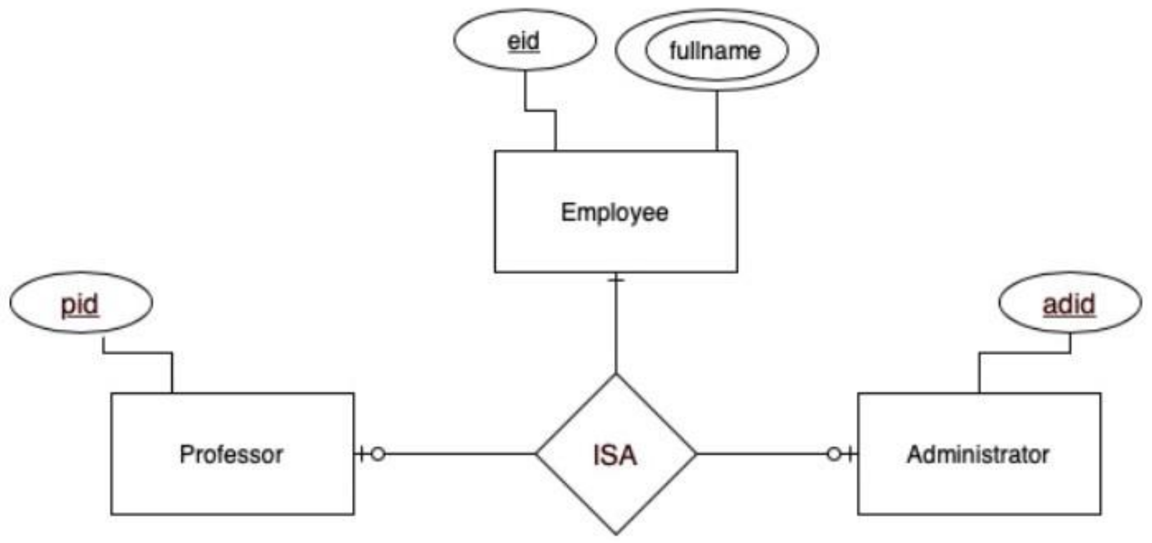
eid and cid are PK in EmployeeByCompany table because we do not want the same employee working for the same company twice when data is inserted in this table. If you wish to have duplicates then remove the PK from eid and cid in table EmployeeByCompany . However, you'll need to query like this when want to retrieve all the employees from a specific company

```
SELECT DISTINCT Employee.fullname FROM Employee
JOIN EmployeeByCompany ON Employee.eid = EmployeeByCompany.eid
JOIN Company ON Company.cid = EmployeeByCompany.cid
WHERE Company.name = "SFSU"
```

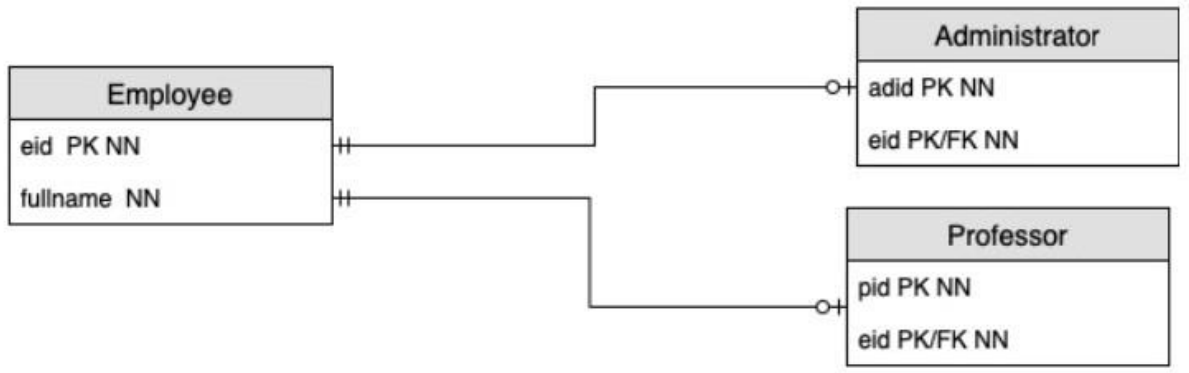
FROM ERD TO EER

ISA

ERD



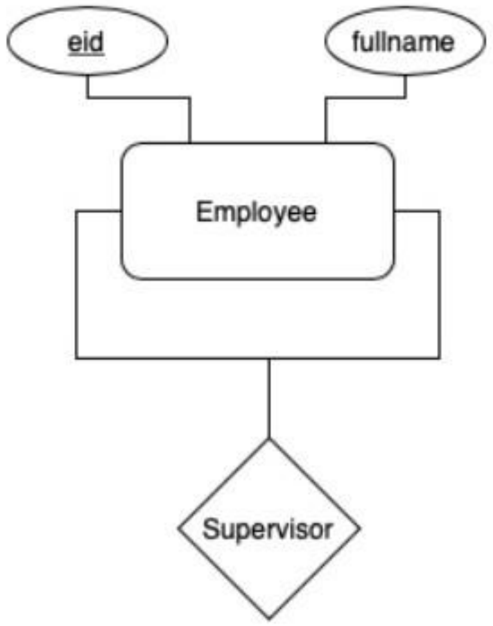
Database Model



FROM ERD TO EER

Recursion when the recursive entity belongs to the same entity set (table)

ERD



Database Model First Way

Employee
eid PK NN
fullaname NN
is_supervisor NN BOOL

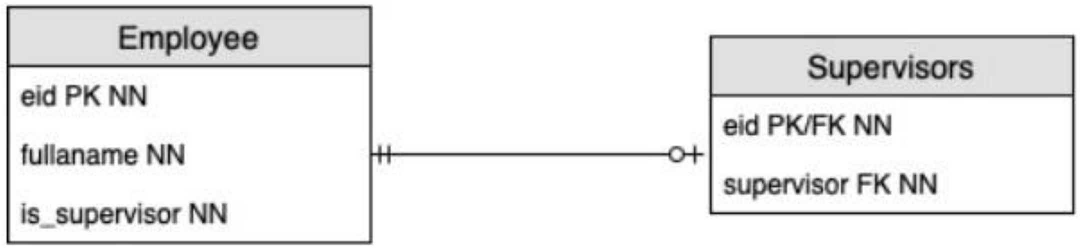
Problem 1: does not define who is the Employee that is being supervised

Database Model Second Way

Employee
eid PK NN
fullaname NN
supervisor FK

Problem 2: supervisor can be NULL because not all the employees must be supervised

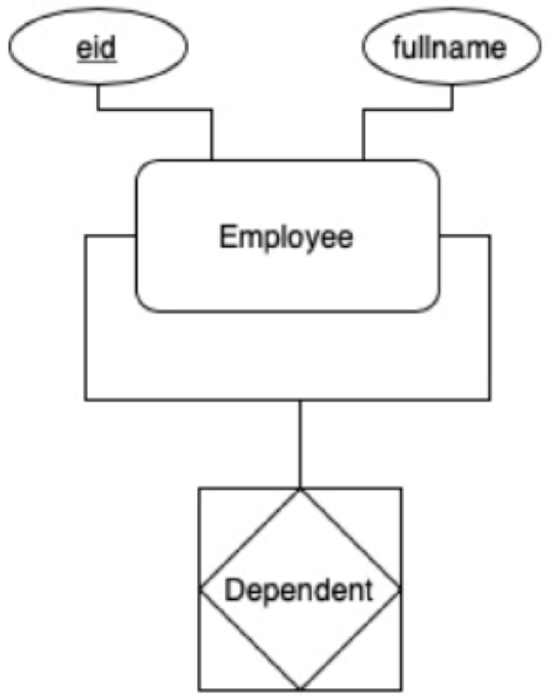
Database Model Third Way



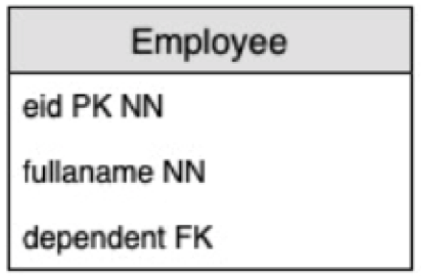
Solves problems 1 and 2

Recursion when the recursive entity belongs to a different entity set (table)

ERD

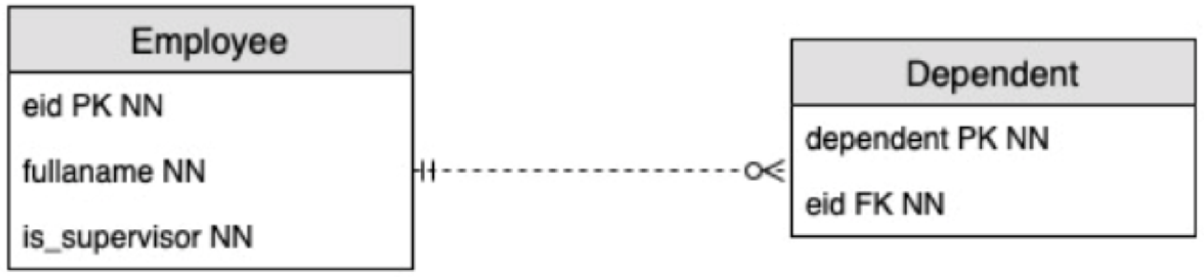


Database Model First Way



Problem 1: What about if we want to add more dependents? Not possible with this configuration
Problem 2: We are enforced to have NULL when the employee has no dependents

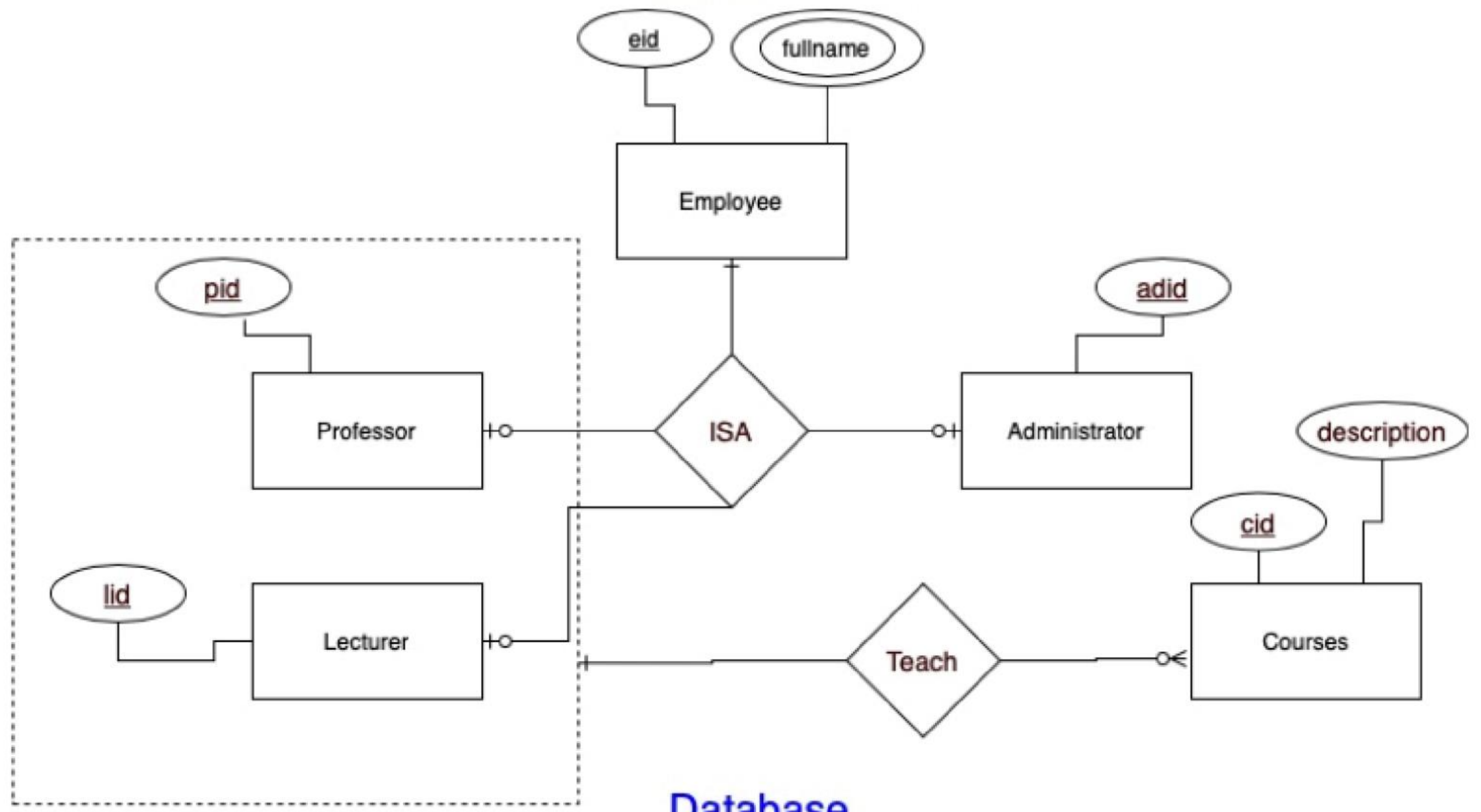
Database Model Second Way



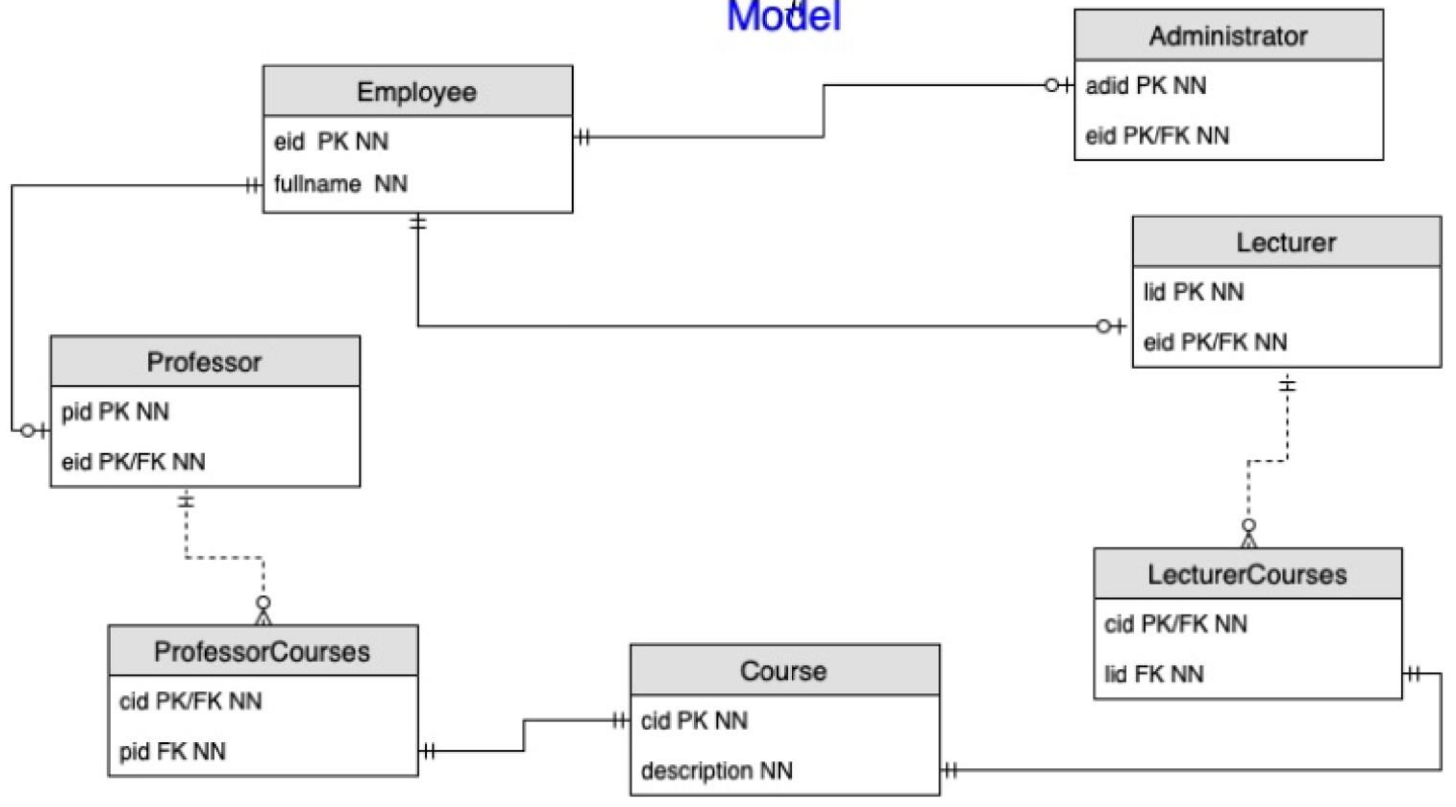
Solves the problem 1 and 2

Aggregation Two Implementations: First one.

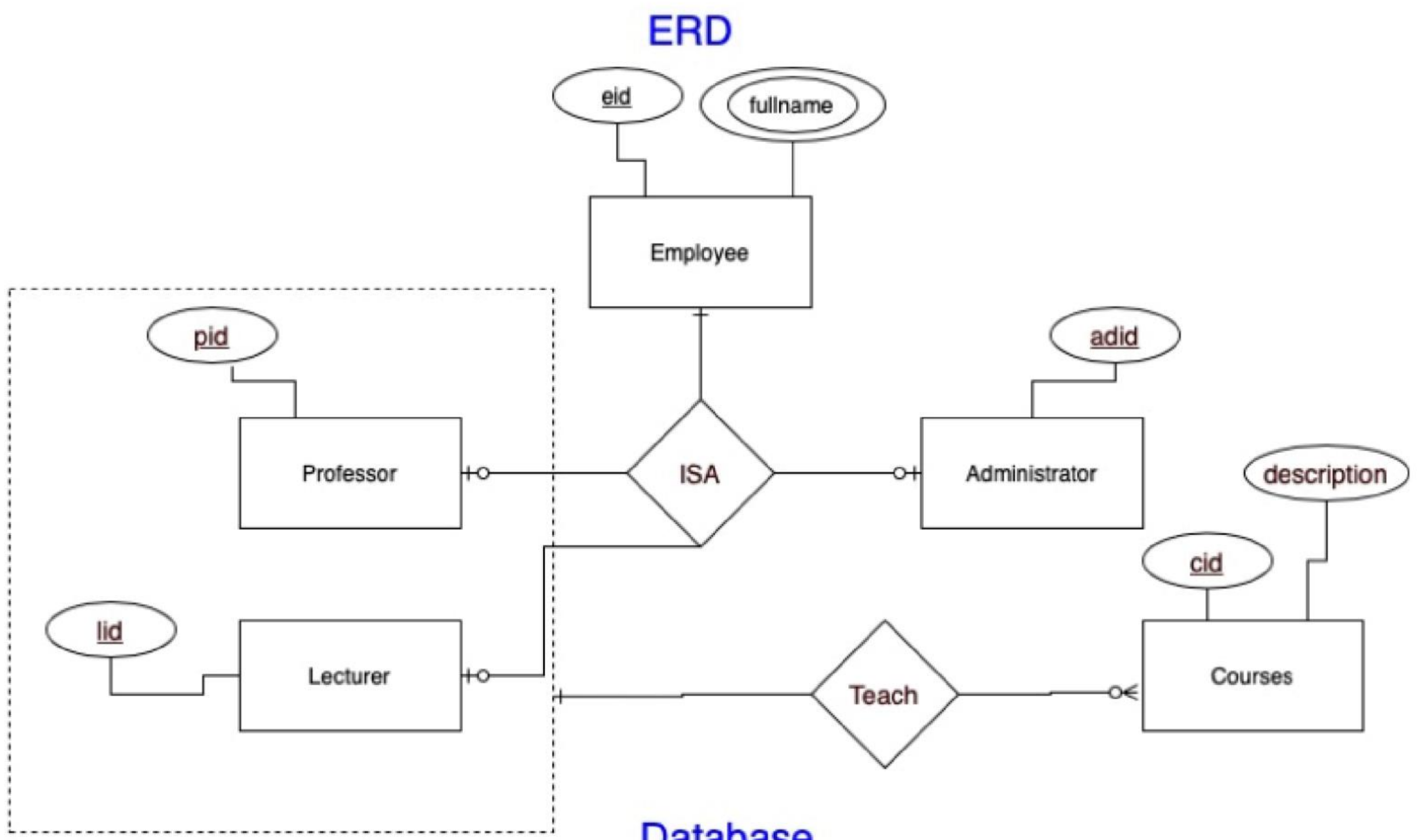
ERD

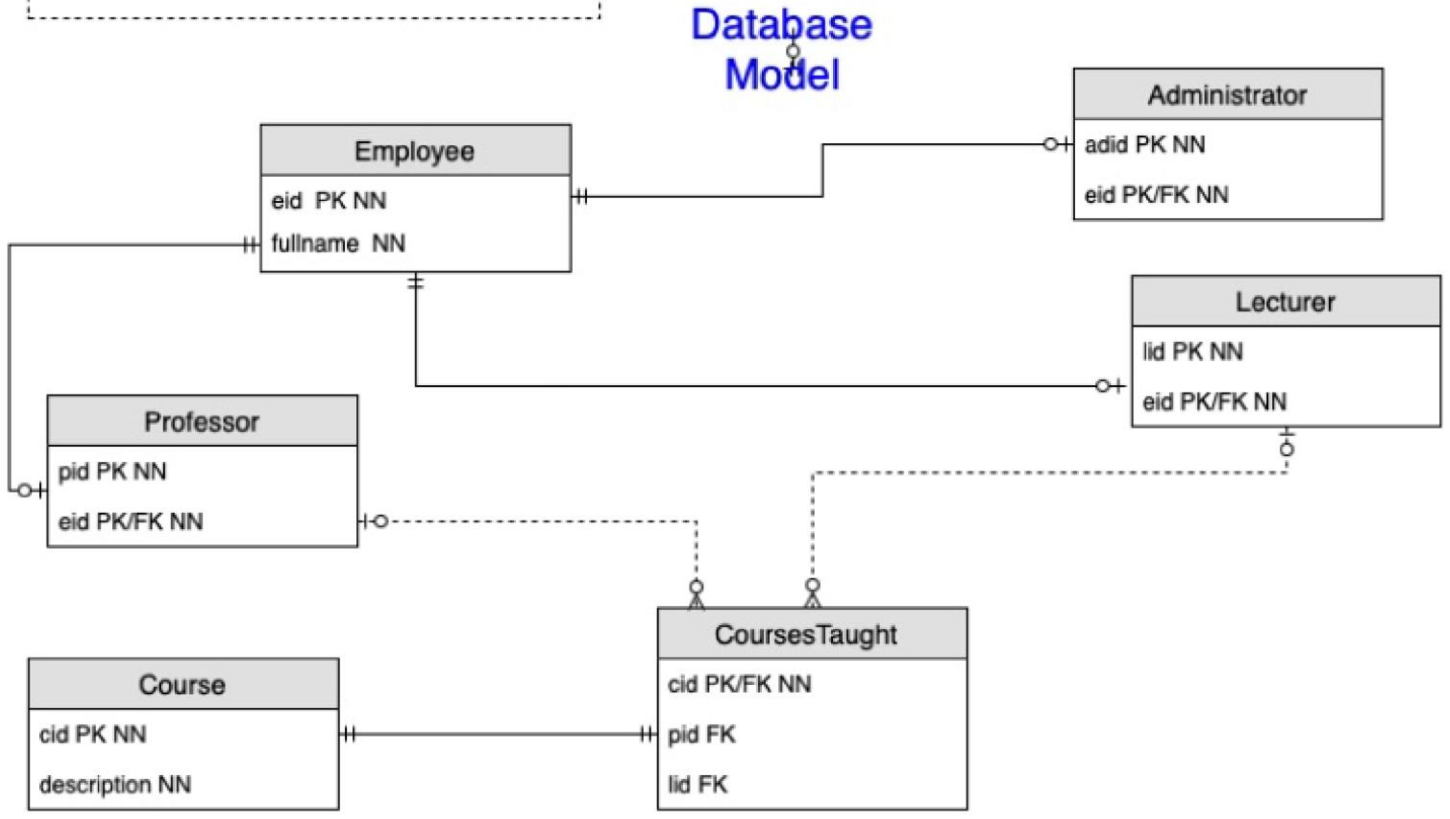


Database Model



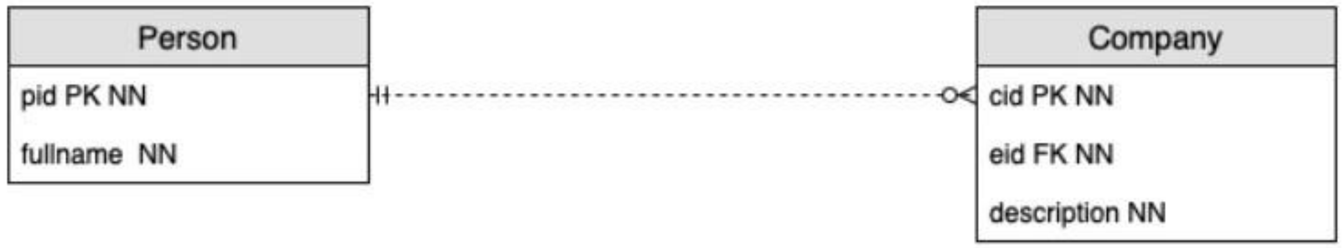
Aggregation Two Implementations: Second one.



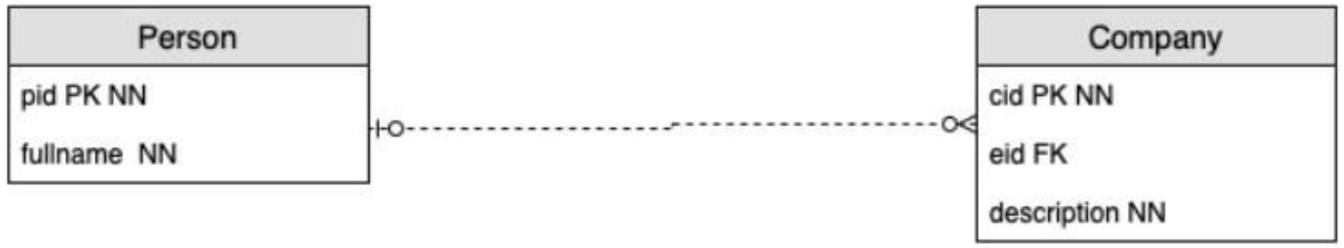


EER: Handling Zeros and Ones

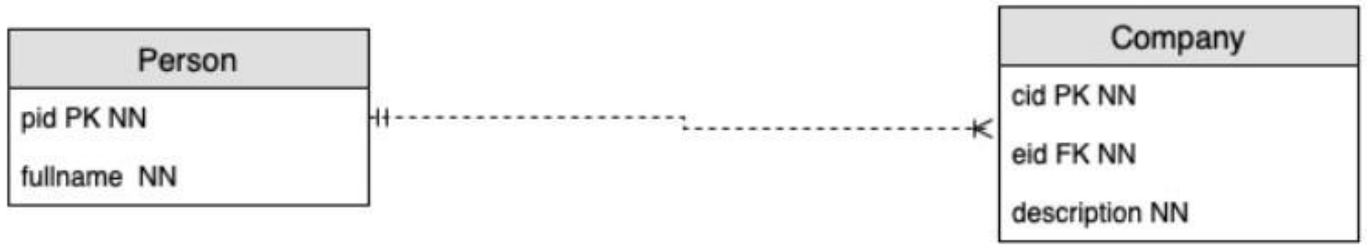
A person entity does not need to be in the company table once it is inserted in the Person table. However once a company entity is inserted in the company table it must have a employee that already exist in the Person table



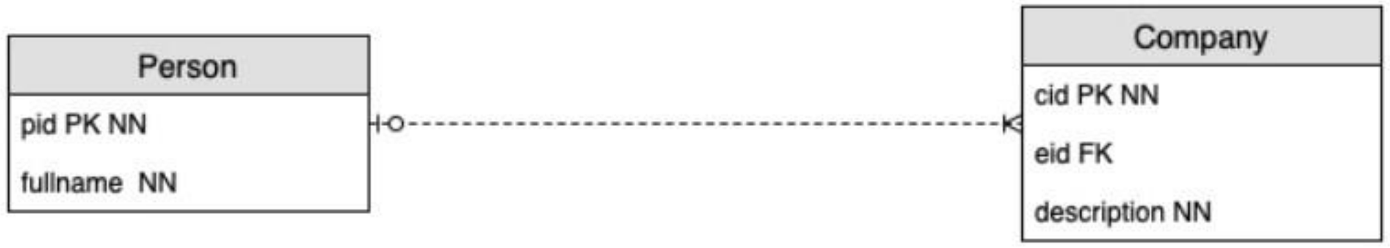
A person entity does not need to be in the company table once it is inserted in the Person table. Also it is not required/not mandatory for a Company entity, once is inserted in the company table, to have a employee



Once a Person entity is inserted in the Person table it also must exist in the Company table, and once a Company entity is inserted in the Company table, it also must have an employee from the Person table



The following configuration is not possible: A Person entity once is inserted in the Person table must be also in the Company table. However, a Company entity once is inserted in a Company table does not need to have an employee from the Person table.



Enhanced Entity-Relationship Model (EER)

- Assigning type of relationship
- Identifying relationship (defined as a solid line)
- The foreign key in a child table can only exist **only and only if** it is totally dependent on the parent table primary key.

- Non identifying relationship (defined as a dashed line) •

  The foreign key of a table referencing its parent table, is not primary key in the child table.

Most Important Domains (data types)

- **Strings:**
- CHAR(range): A FIXED length string (can contain letters, numbers, and special characters). 0 to 255
- VARCHAR(range): A VARIABLE length string (can contain letters, numbers, and special characters). 0 to 65535
- TEXT(size): Holds a string with a maximum length of 65,535 bytes

Most Important Domains

- BLOB(size): For BLOBs (Binary Large OBjects). Holds up to 65,535 bytes of data

(data types)

- **Numeric:**

- BIT(range): A bit-value type. The number of bits per value is specified in *size.* 0 to 64

- TINYINT(range): A very small integer. Signed range is from -128 to 127. Unsigned range?

Most Important Domains

- BOOL(size): Zero is considered as false, nonzero values are considered as true.

- INT(size): A medium integer. Signed range is from -2147483648 to 2147483647. Unsigned range is from 0 to 4294967295.

- DECIMAL(x,y): for numbers with fractions requiring exact precision. Can be signed (allow positive and negative values) or unsigned (only allow positive numbers). Commonly used for monetary fields.

(data types)

- **Date and Times:**

Most Important Domains

- DATE: A date. Format: YYYY-MM-DD. The supported range is from '1000-01-01' to '9999-12-31'

- DATETIME(fsp): A date and time combination. Format: YYYY-MM-DD hh:mm:ss.

- TIMESTAMP(fsp): A timestamp. TIMESTAMP values are stored as the number of seconds since the Unix epoch ('1970-01-01 00:00:00' UTC). Format: YYYYMM-DD hh:mm:ss.

- TIME: A time. Format: hh:mm:ss. The supported range is from '-838:59:59' to ‘838:59:59'

Most Important Domains

- YEAR: A year in four-digit format. Values allowed in four-digit format: 1901 to 2155, and 0000. MySQL 8.0 does not support year in two-digit format.

How to choose the right data type?

- Is choosing ‘smaller' datatypes usually better for memory performance and disk storage?
- Choosing between CHAR or VARCHAR?
- Choosing between TYNINT or any other INT type for keys?
- Choosing between numeric or alphanumeric types for keys?
- Is there any memory performance improvement when choosing between unsigned and signed types?

How to choose the right data type?

- What’s the real meaning of NULL in databases. Should we avoid NULL when possible?
- Be careful when choosing random strings for your primary keys

Here is one example of two similar strings "*aaa*" and "*aab*" and their SHA1 values "*7e240de74fb1ed08fa08d38063f6a6a91462a815*", "*40b904fd8852297daeaeb426b1bca46fd2454aa3*"

How to choose the right data type?

respectively. Although strings "*aaa*" and "*aab*" are very similar, their SHA1 results are not and this can have a big impact on INSERT statements, because new row has to go into random location in index. This can cause page splits and random disk access which can slow overall performance.

- The issues we encounter with the data types MySQL automatically selected are:

1. Wrong data type:

- String in time operations

How to choose the right data type?

2. Too constrained

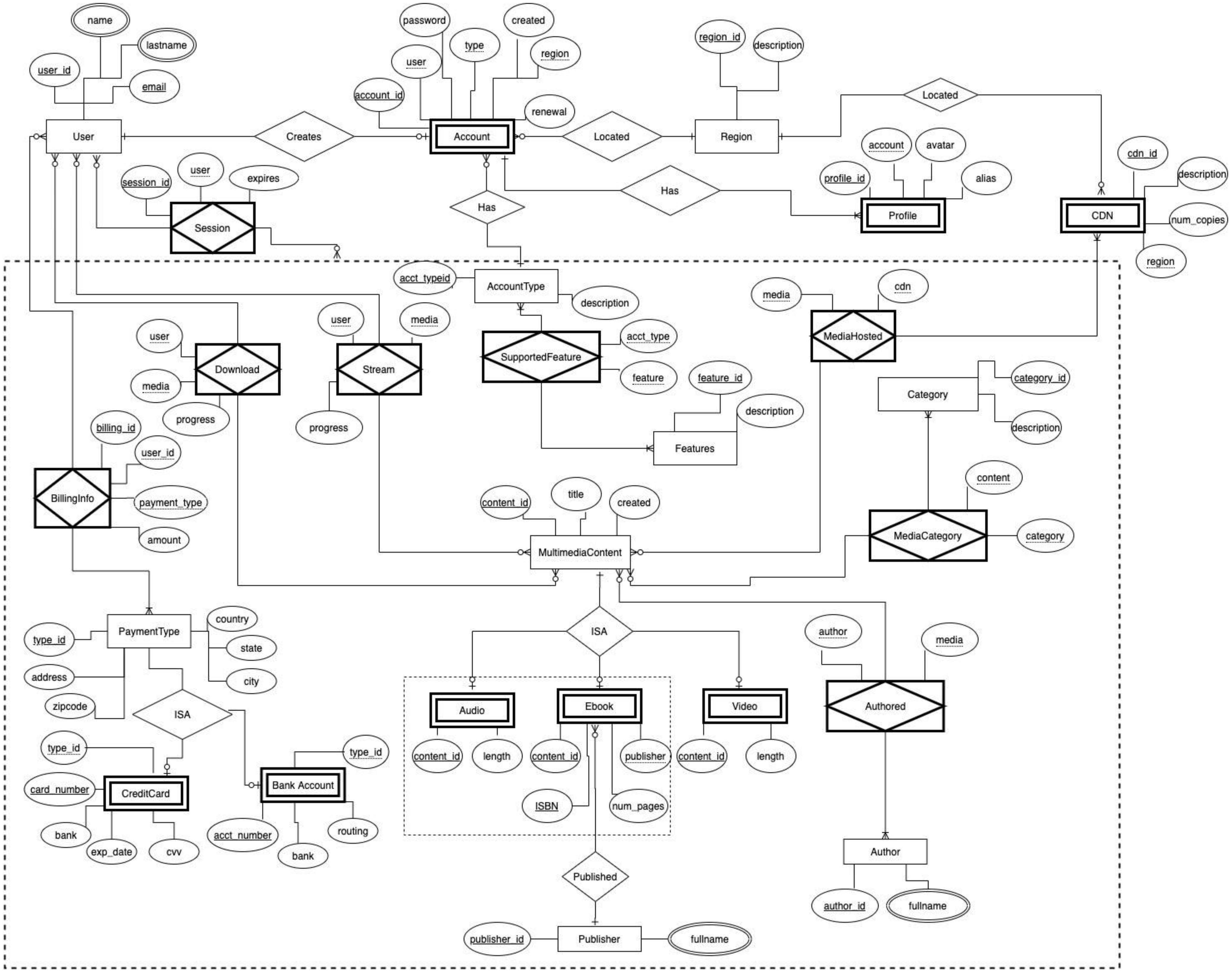
- Larger values than the data type supports

3. Too conservative:

- Wasting storage space

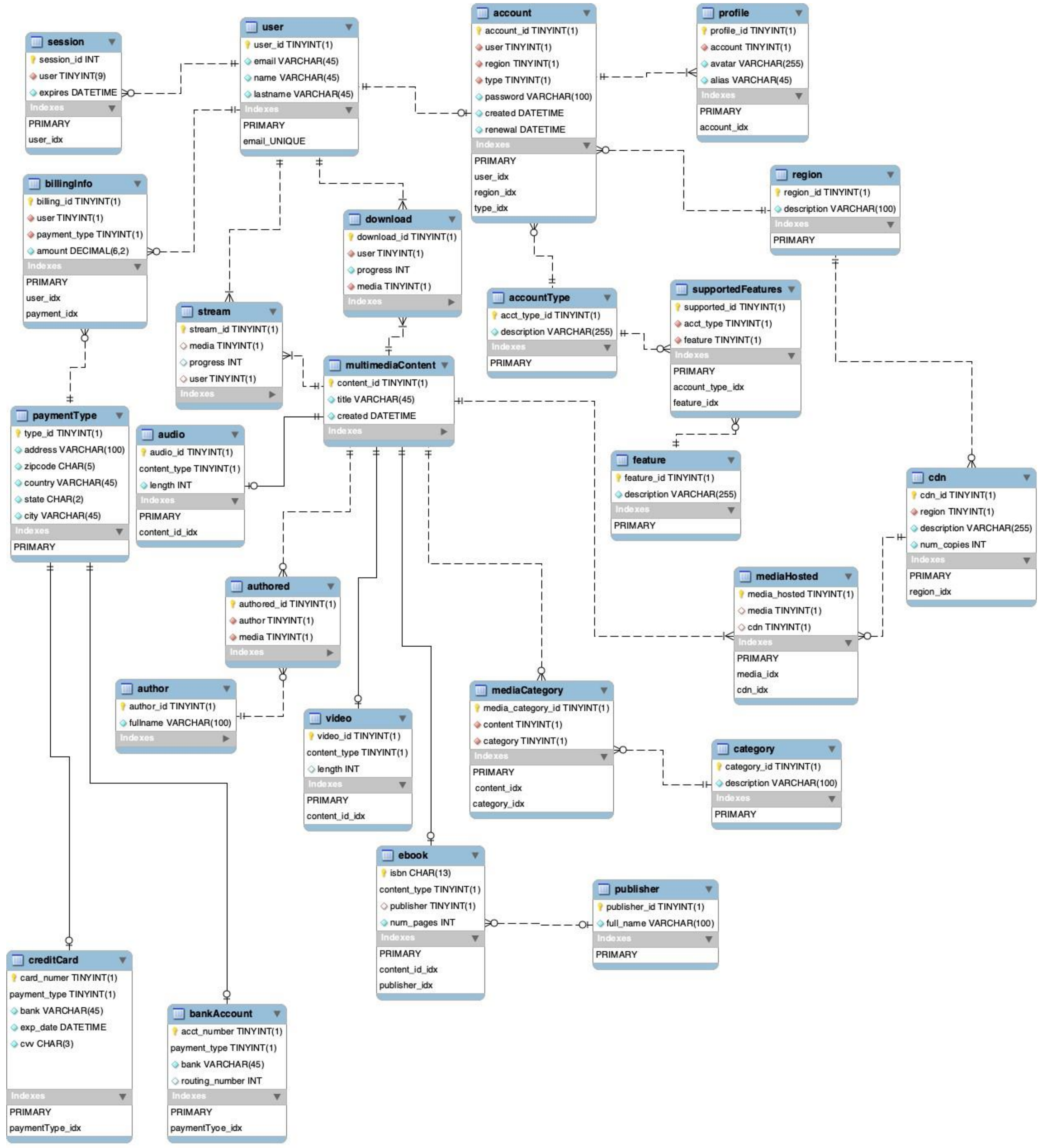
Building an EER Multimedia Streaming Service Database System

Multimedia Streaming Service Database System ERD (This is an optimized version)



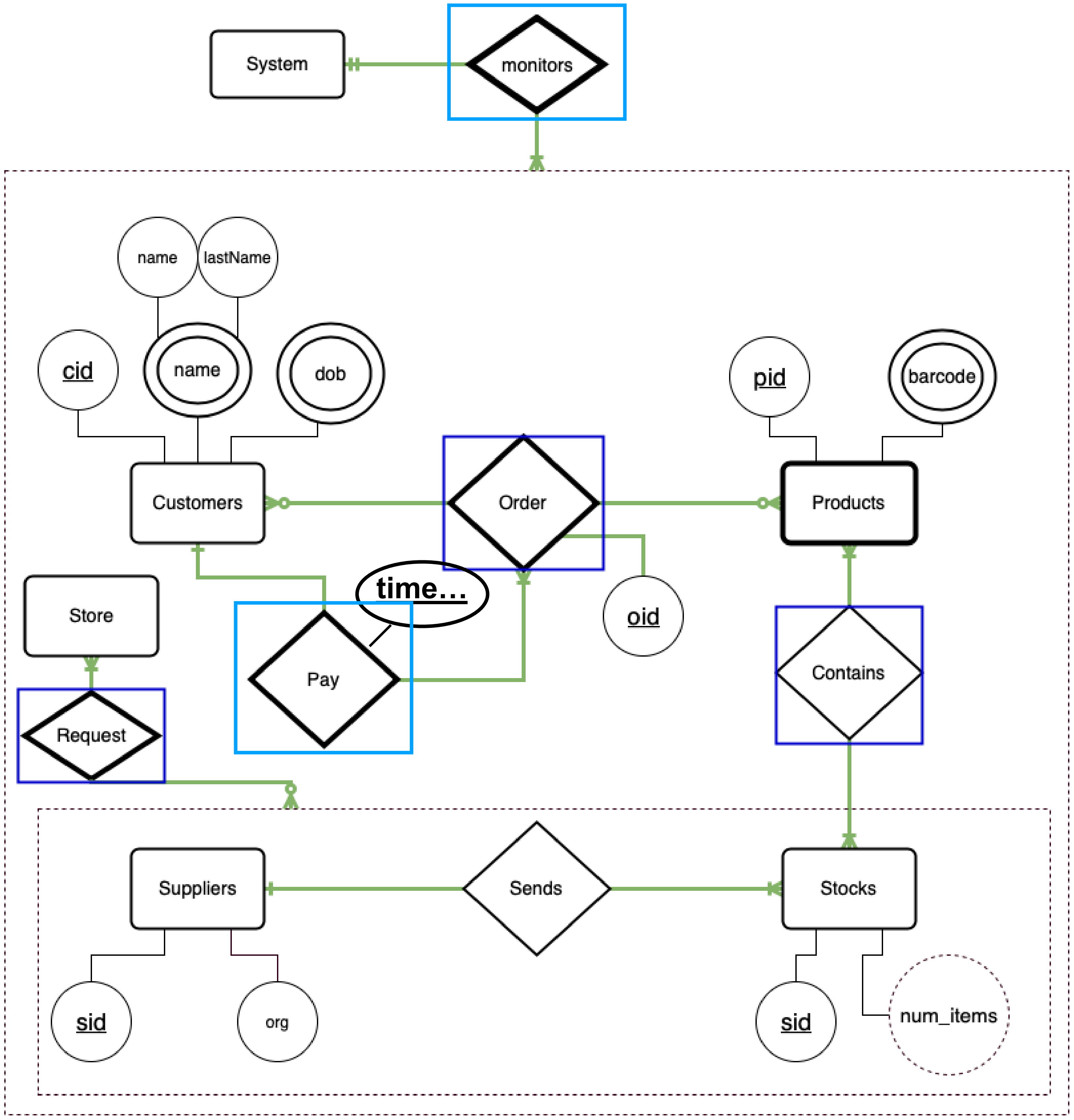
name
lastname
user_id
email
User
Creates
password
created
type
user
region
account_id
Account
renewal
Located
Region
region_id
description
Located
account
avatar
profile_id
alias
Profile
Has
Has
cdn_id
description
CDN
num_copies
region
session_id
user
expires
Session
acct_typeid
AccountType
description
SupportedFeature
acct_type
feature
feature_id
description
Features
media
cdn
MediaHosted
Category
category_id
description
user
Download
media
progress
user
media
Stream
progress
billing_id
user_id
BillingInfo
payment_type
amount
content_id
title
created
MultimediaContent
content
MediaCategory
category
PaymentType
type_id
country
state
city
address
zipcode
ISA
type_id
CreditCard
card_number
bank
exp_date
cvv
Bank Account
type_id
acct_number
bank
routing
ISA
Audio
content_id
length
Ebook
content_id
publisher
ISBN
num_pages
Video
content_id
length
author
media
Authored
Published
publisher_id
Publisher
fullname
Author
author_id
fullname

Multimedia Streaming Service Database Model (EER)

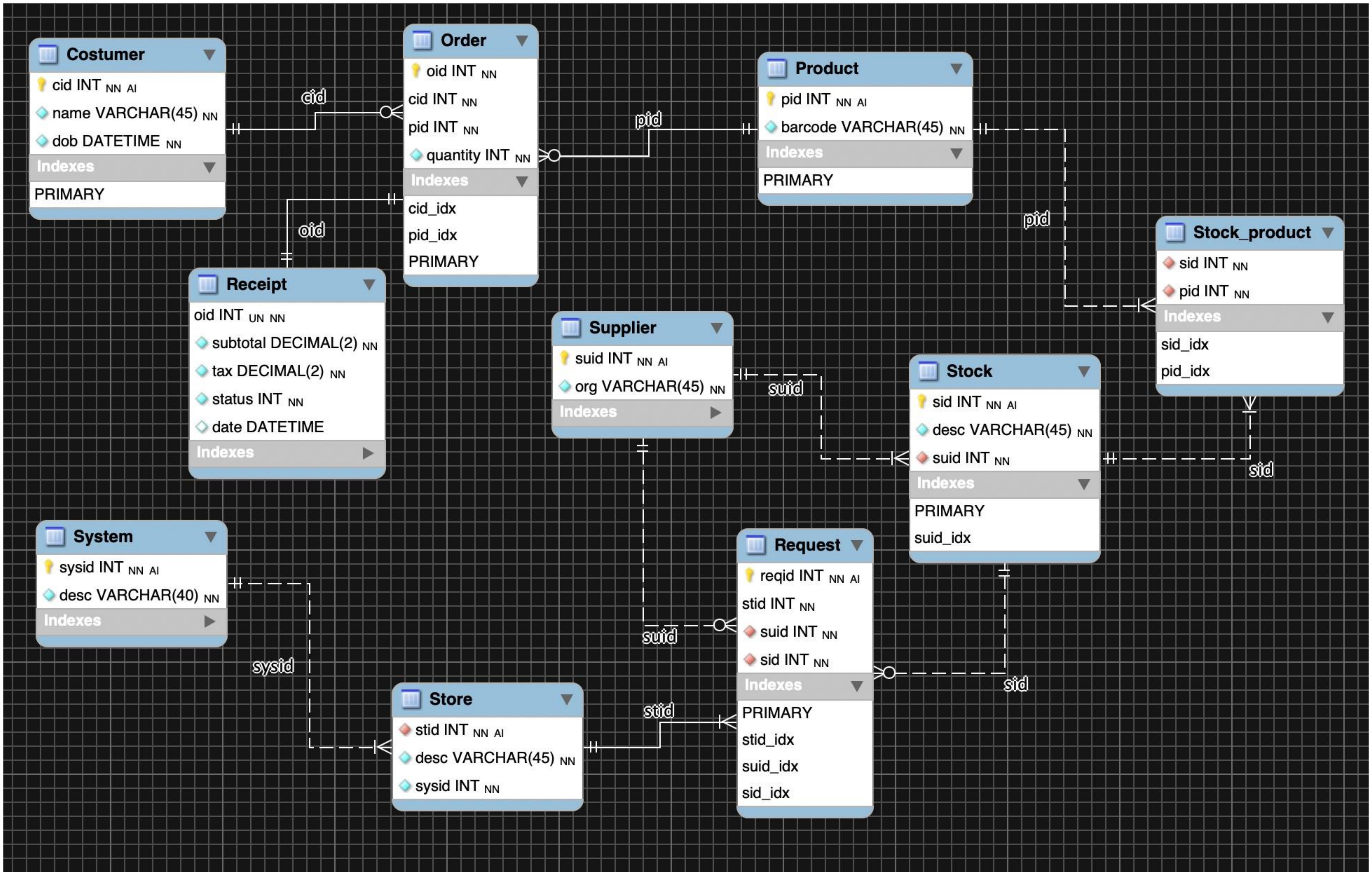

session
session_id INT
user TINYINT(9)
expires DATETIME
Indexes
PRIMARY
user_idx
user
user_id TINYINT(1)
email VARCHAR(45)
name VARCHAR(45)
lastname VARCHAR(45)
Indexes
PRIMARY
email_UNIQUE
account
account_id TINYINT(1)
user TINYINT(1)
region TINYINT(1)
type TINYINT(1)
password VARCHAR(100)
created DATETIME
renewal DATETIME
Indexes
PRIMARY
user_idx
region_idx
type_idx
profile
profile_id TINYINT(1)
account TINYINT(1)
avatar VARCHAR(255)
alias VARCHAR(45)
Indexes
PRIMARY
account_idx
region
region_id TINYINT(1)
description VARCHAR(100)
Indexes
PRIMARY
billingInfo
billing_id TINYINT(1)
user TINYINT(1)
payment_type TINYINT(1)
amount DECIMAL(6,2)
Indexes
PRIMARY
user_idx
payment_idx
download
download_id TINYINT(1)
user TINYINT(1)
progress INT
media TINYINT(1)
Indexes
accountType
acct_type_id TINYINT(1)
description VARCHAR(255)
Indexes
PRIMARY
supportedFeatures
supported_id TINYINT(1)
acct_type TINYINT(1)
feature TINYINT(1)
Indexes
PRIMARY
account_type_idx
feature_idx
stream
stream_id TINYINT(1)
media TINYINT(1)
progress INT
user TINYINT(1)
Indexes
multimediaContent
content_id TINYINT(1)
title VARCHAR(45)
created DATETIME
Indexes
paymentType
type_id TINYINT(1)
address VARCHAR(100)
zipcode CHAR(5)
country VARCHAR(45)
state CHAR(2)
city VARCHAR(45)
Indexes
PRIMARY
audio
audio_id TINYINT(1)
content_type TINYINT(1)
length INT
Indexes
PRIMARY
content_id_idx
feature
feature_id TINYINT(1)
description VARCHAR(255)
Indexes
PRIMARY
cdn
cdn_id TINYINT(1)
region TINYINT(1)
description VARCHAR(255)
num_copies INT
Indexes
PRIMARY
region_idx
mediaHosted
media_hosted TINYINT(1)
media TINYINT(1)
cdn TINYINT(1)
Indexes
PRIMARY
media_idx
cdn_idx
authored
authored_id TINYINT(1)
author TINYINT(1)
media TINYINT(1)
Indexes
author
author_id TINYINT(1)
fullname VARCHAR(100)
Indexes
video
video_id TINYINT(1)
content_type TINYINT(1)
length INT
Indexes
PRIMARY
content_id_idx
mediaCategory
media_category_id TINYINT(1)
content TINYINT(1)
category TINYINT(1)
Indexes
PRIMARY
content_idx
category_idx
category
category_id TINYINT(1)
description VARCHAR(100)
Indexes
PRIMARY
ebook
isbn CHAR(13)
content_type TINYINT(1)
publisher TINYINT(1)
num_pages INT
Indexes
PRIMARY
content_id_idx
publisher_idx
publisher
publisher_id TINYINT(1)
full_name VARCHAR(100)
Indexes
PRIMARY
creditCard
card_numer TINYINT(1)
payment_type TINYINT(1)
bank VARCHAR(45)
exp_date DATETIME
cvv CHAR(3)
Indexes
PRIMARY
paymentType_idx
bankAccount
acct_number TINYINT(1)
payment_type TINYINT(1)
bank VARCHAR(45)
routing_number INT
Indexes
PRIMARY
paymentTyoe_idx

Let’s Build another EER For Ordering System

Final ERD


System
monitors
name
lastName
cid
name
dob
pid
barcode
Customers
Order
Products
Store
time…
oid
Pay
Contains
Request
Suppliers
Sends
Stocks
sid
org
sid
num_items

Database Model (EER)


Costumer
cid INT NN AI
name VARCHAR(45) NN
dob DATETIME NN
Indexes
PRIMARY
Order
oid INT NN
cid INT NN
pid INT NN
quantity INT NN
Indexes
cid_idx
pid_idx
PRIMARY
Product
pid INT NN AI
barcode VARCHAR(45) NN
Indexes
PRIMARY
Stock_product
sid INT NN
pid INT NN
Indexes
sid_idx
pid_idx
Receipt
oid INT UN NN
subtotal DECIMAL(2) NN
tax DECIMAL(2) NN
status INT NN
date DATETIME
Indexes
Supplier
suid INT NN AI
org VARCHAR(45) NN
Indexes
Stock
sid INT NN AI
desc VARCHAR(45) NN
suid INT NN
Indexes
PRIMARY
suid_idx
System
sysid INT NN AI
desc VARCHAR(40) NN
Indexes
Request
reqid INT NN AI
stid INT NN
suid INT NN
sid INT NN
Indexes
PRIMARY
stid_idx
suid_idx
sid_idx
Store
stid INT NN AI
desc VARCHAR(45) NN
sysid INT NN
cid
oid
pid
pid
suid
sid
suid
sid
sysid
stid