Data Management and Warehousing

Tutorial 2: ER-Modelling

Biswadeep Sen
School of Computing
National University of Singapore
biswadeep@u.nus.edu

The problem:

The Varsity International Network of Oenology wishes to computerise the management of the information about its members as well as to record the information they gather about various wines. Your company, Apasaja Private Limited, is commissioned by the Varsity International Network of Oenology to design and implement the relational schema of the database application. The organisation is big enough so that there could be several members with the same name. A card with a unique number is issued to identify each drinker. The contact address of each member is also recorded for the mailing of announcements and calls for meetings.

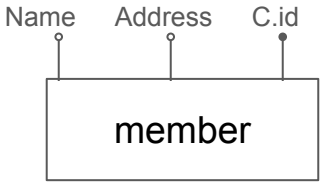
At most once a week, VINO organises a tasting session. At each session, the attending members taste several bottles. Each member records for each bottle his or her evaluation of the quality (very good, good, average, mediocre, bad, very bad) of each wine that she or he tastes. The evaluation may differ for the same wine from one drinker to another. Actual quality and therefore evaluation also varies from one to another bottle of a given wine. Every bottle that is opened during the tasting session is finished during that session.

Each wine is identified by its name (“Parade D'Amour”), appellation (“Bordeaux”) and vintage (1990). Other information of interest about the wine is the degree of alcohol (11.5), where and by whom it has been bottled (“Mis en Bouteille par Amblard-Larolphie Negociant-Eleveur a Saint Andre de Cubzac (Gironde) - France”), the certification of its appellation if available (“Appellation Bordeaux Controlée”), and the country it comes from (produce of “France”).

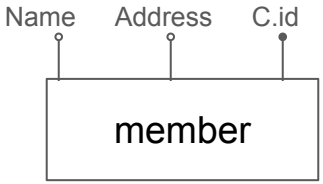
Generally, there are or have been several bottles of the same wine in the cellar. For each wine, the bottles in the wine cellar of VINO are numbered. For instance, the cellar has 20 bottles numbered 1 to 20 of a Semillon from 1996 named Rumbalara. For documentation purposes, VINO may also want to record wines for which it does not own bottles. The bottles are either available in the cellar or they have been tasted and emptied.

We first want to design an entity-relationship schema that most correctly and most completely captures the constraints expressed in the above description of the VINO application.

“The organisation is big enough so that there could be several **members** with the same **name**. **A card with a unique number is issued to identify each drinker.** The **contact address** of each member is also recorded for the mailing of announcements and calls for meetings.”



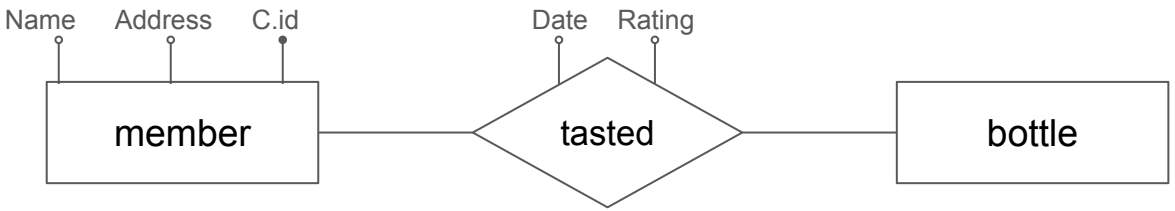
“The organisation is big enough so that there could be several **members** with the same **name**. **A card with a unique number is issued to identify each drinker.** The **contact address** of each member is also recorded for the mailing of announcements and calls for meetings.”



member is an entity set with attributes: **Name**, **Address** and **Card Number (C.id)**, where C.id acts as the primary key.

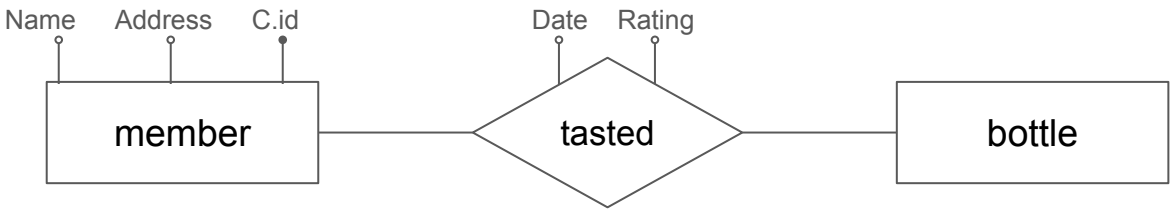
“At most once a week, VINO organises a tasting session. At each session, the **attending members** taste **several bottles**. Each member records for each bottle his or her **evaluation** of the quality (very good, good, average, mediocre, bad, very bad) of each wine that she or he tastes. The evaluation may differ for the same wine from one drinker to another. Actual quality and therefore evaluation also varies from one to another bottle of a given wine. Every bottle that is opened during the tasting session is finished during that session.”

Bottles is another entity set, and is related to **member** by a relationship called **tasted**: **Members taste bottles** in different sessions to give **ratings**.



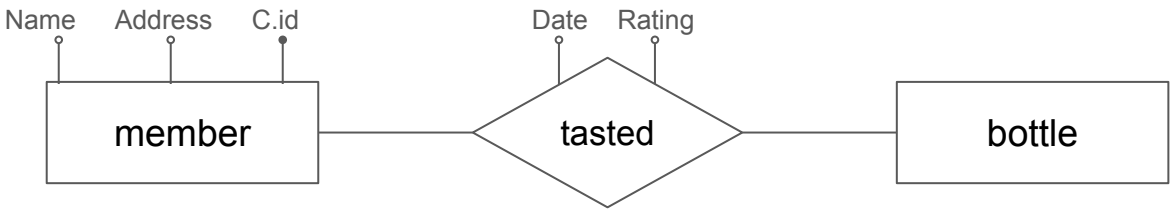
“At most once a week, VINO organises a tasting session. At each session, the **attending members** taste **several bottles**. Each member records for each bottle his or her **evaluation** of the quality (very good, good, average, mediocre, bad, very bad) of each wine that she or he tastes. The evaluation may differ for the same wine from one drinker to another. Actual quality and therefore evaluation also varies from one to another bottle of a given wine. Every bottle that is opened during the tasting session is finished during that session.”

bottle is another entity set, and is related to **member** by a relationship called **tasted**: **Members taste bottles** in different sessions to give **ratings**.

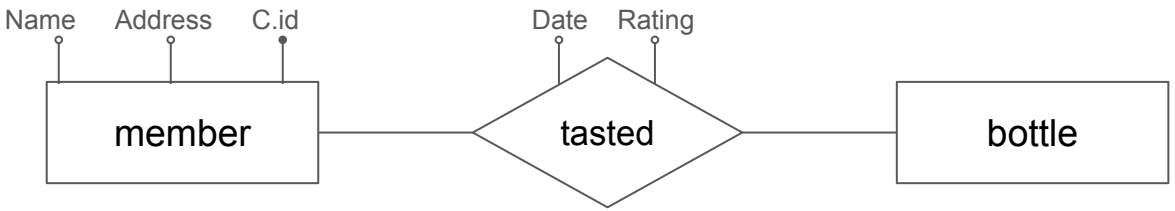


We do not yet know the attributes of the **bottle** entity. But essentially necessary attributes from **member** and **bottle** will be borrowed by the relationship **tasted.**

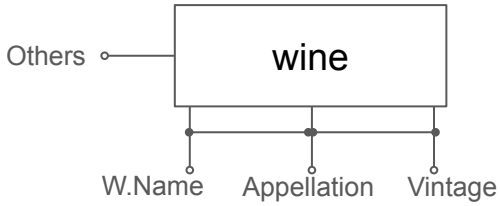
“Each **wine** is **identified by its name (“Parade D'Amour”), appellation (“Bordeaux”) and vintage (1990)**. Other information of interest about the wine is the **degree of alcohol** (11.5), where and **by whom it has been bottled** (“Mis en Bouteille par Amblard-Larolphie Negociant-Eleveur a Saint Andre de Cubzac (Gironde) - France”), the **certification of its appellation** if available (“Appellation Bordeaux Controlée”), and the **country it comes from** (produce of “France”).”



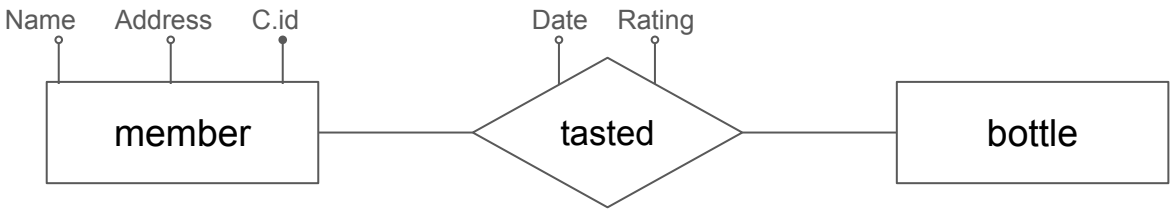
“Each **wine** is **identified by its name (“Parade D'Amour”), appellation (“Bordeaux”) and vintage (1990)**. Other information of interest about the wine is the **degree of alcohol** (11.5), where and **by whom it has been bottled** (“Mis en Bouteille par Amblard-Larolphie Negociant-Eleveur a Saint Andre de Cubzac (Gironde) - France”), the **certification of its appellation** if available (“Appellation Bordeaux Controlée”), and the **country it comes from** (produce of “France”).”



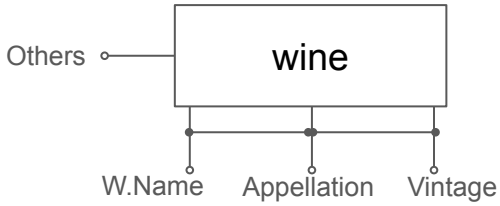
wine is another entity set with a **composite primary key** which consists of **Name, Appellation & Vintage.** It has other attributes: Alcohol_degree, Bottled_by, Appellation_cert and Country.



“Each **wine** is **identified by its name (“Parade D'Amour”), appellation (“Bordeaux”) and vintage (1990)**. Other information of interest about the wine is the **degree of alcohol** (11.5), where and **by whom it has been bottled** (“Mis en Bouteille par Amblard-Larolphie Negociant-Eleveur a Saint Andre de Cubzac (Gironde) - France”), the **certification of its appellation** if available (“Appellation Bordeaux Controlée”), and the **country it comes from** (produce of “France”).”

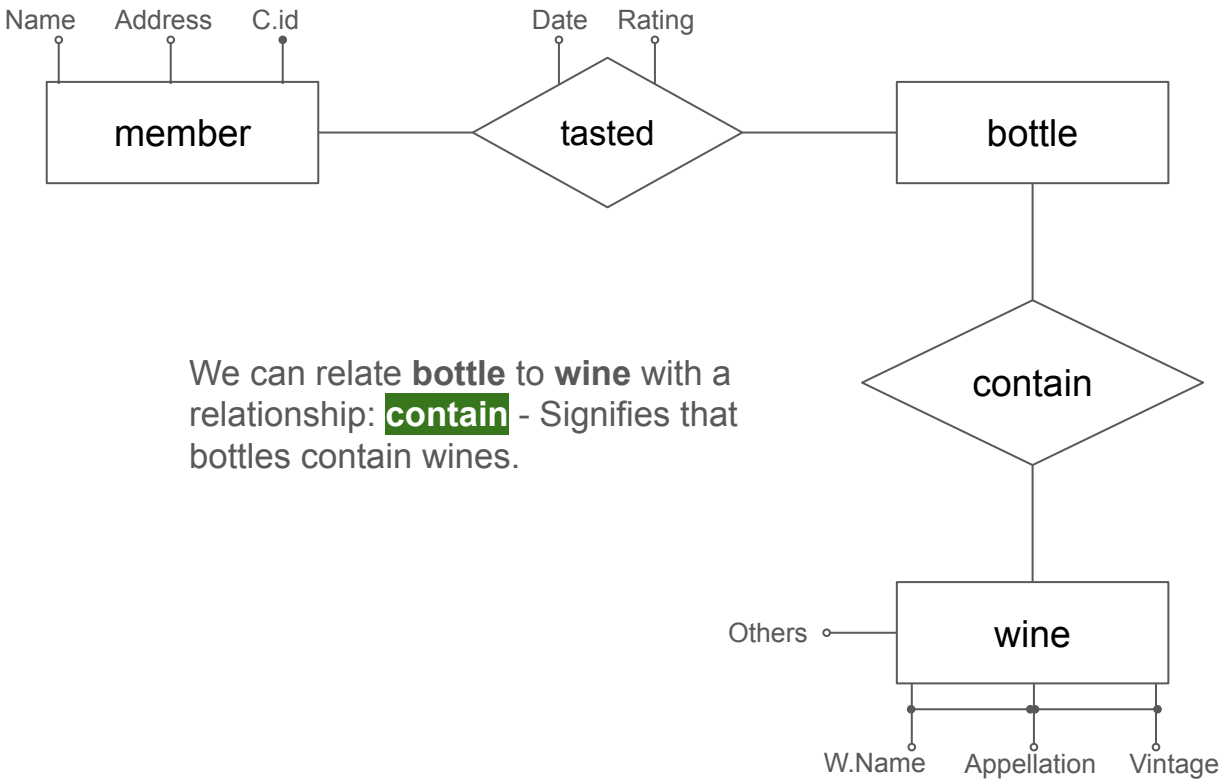


wine is another entity set with a **composite primary key** which consists of **Name, Appellation & Vintage.** It has other attributes: Alcohol_degree, Bottled_by, Appellation_cert and Country.



"Each **wine** is **identified by its name ("Parade D'Amour"), appellation ("Bordeaux") and vintage (1990)**. Other information of interest about the wine is the **degree of alcohol** (11.5), where and **by whom it has been bottled** ("Mis en Bouteille par Amblard-Larolphie Negociant-Eleveur a Saint Andre de Cubzac (Gironde) - France"), the **certification of its appellation** if available ("Appellation Bordeaux Controlée"), and the **country it comes from** (produce of "France")."

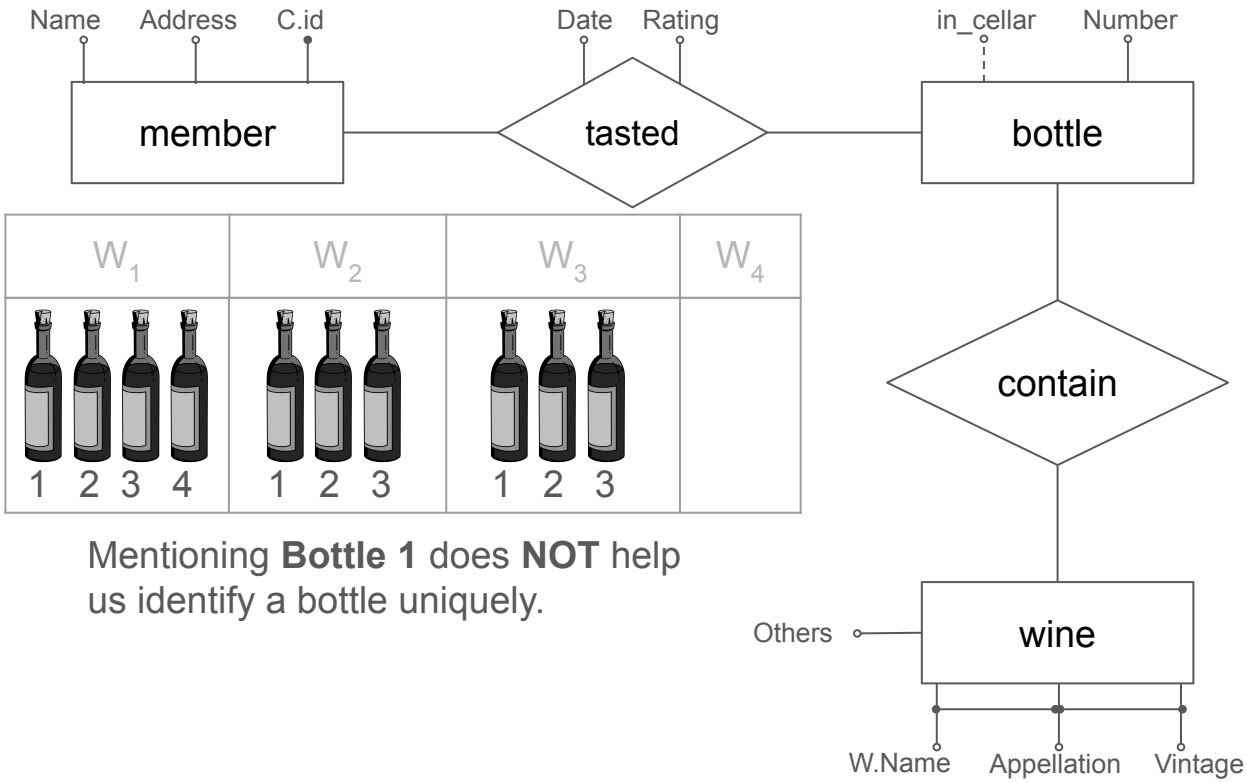
wine is another entity set with a **composite primary key** which consists of **Name, Appellation & Vintage.** It has other attributes: Alcohol_degree, Bottled_by, Appellation_cert and Country (denoted by **Others**).



We can relate **bottle** to **wine** with a relationship: **contain** - Signifies that bottles contain wines.

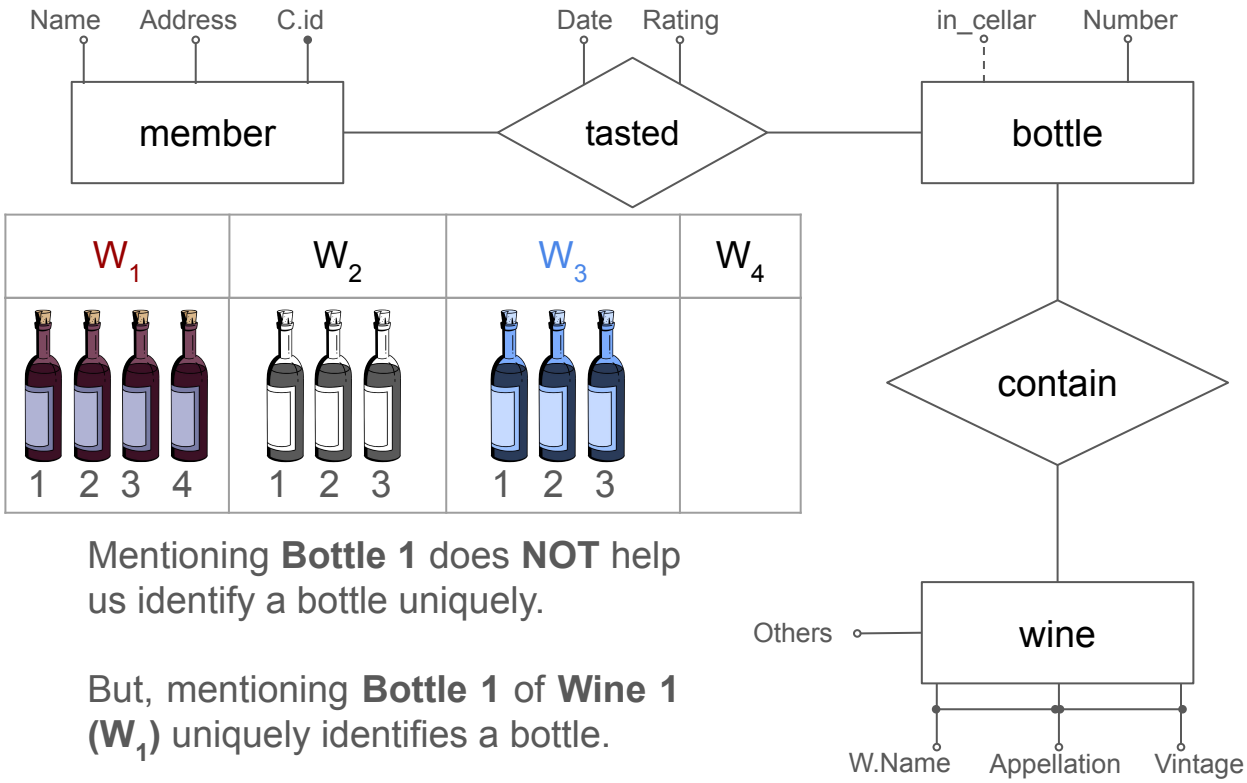
“Generally, there are or have been **several bottles of the same wine** in the cellar. **For each wine, the bottles in the wine cellar of VINO are numbered.** For instance, the cellar has 20 bottles numbered 1 to 20 of a Semillon from 1996 named Rumbalara. **For documentation purposes, VINO may also want to record wines for which it does not own bottles.** The bottles are **either available in the cellar or they have been tasted and emptied**.”

Bottles have **Number** and **Status** as its own attributes.
But these are not enough to identify the bottles **UNIQUELY**.



"Generally, there are or have been **several bottles of the same wine** in the cellar. **For each wine, the bottles in the wine cellar of VINO are numbered.** For instance, the cellar has 20 bottles numbered 1 to 20 of a Semillon from 1996 named Rumbalara. **For documentation purposes, VINO may also want to record wines for which it does not own bottles.** The bottles are **either available in the cellar or they have been tasted and emptied**."

Bottles have **Number** and **Status** as its own attributes. But these are not enough to identify the bottles **UNIQUELY**.

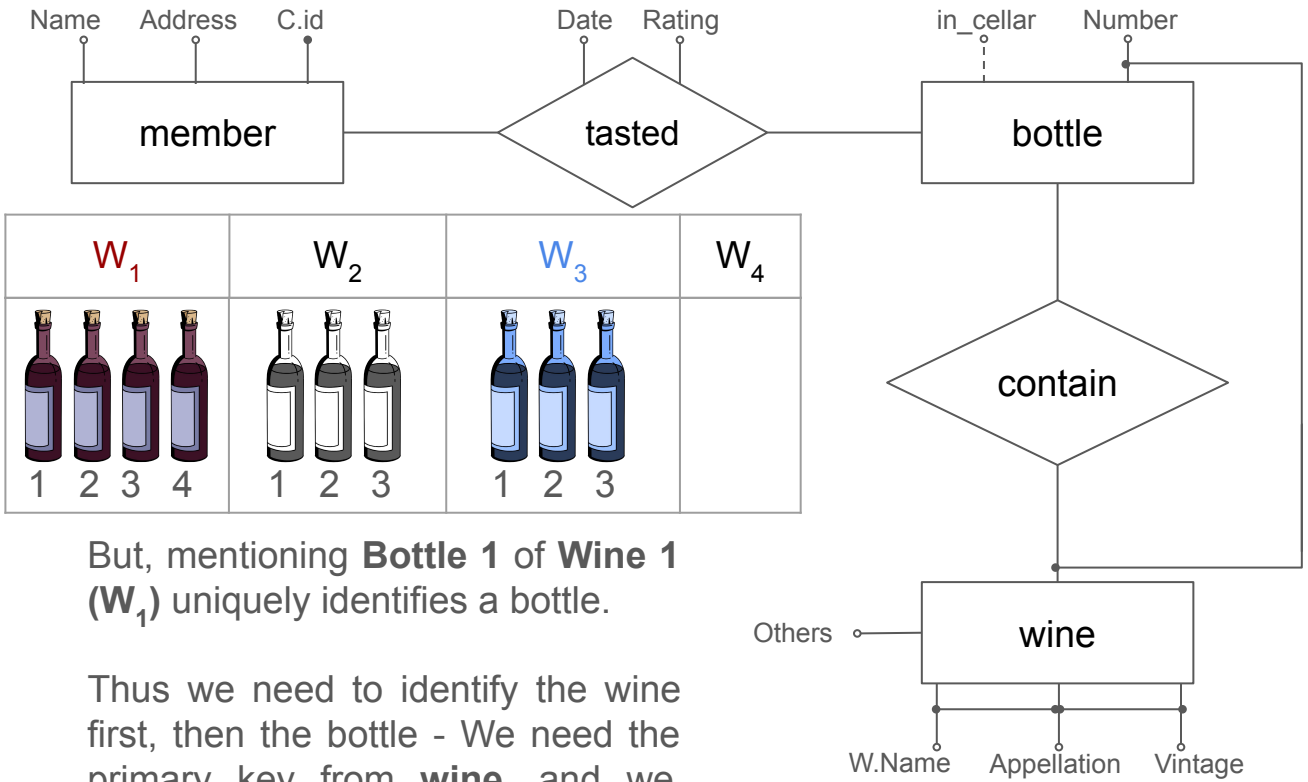


Mentioning **Bottle 1** does **NOT** help us identify a bottle uniquely.

But, mentioning **Bottle 1** of **Wine 1 (W_1)** uniquely identifies a bottle.

"Generally, there are or have been **several bottles of the same wine** in the cellar. **For each wine, the bottles in the wine cellar of VINO are numbered.** For instance, the cellar has 20 bottles numbered 1 to 20 of a Semillon from 1996 named Rumbalara. **For documentation purposes, VINO may also want to record wines for which it does not own bottles.** The bottles are **either available in the cellar or they have been tasted and emptied**."

Bottles have **Number** and **Status** as its own attributes.
But these are not enough to identify the bottles **UNIQUELY**.

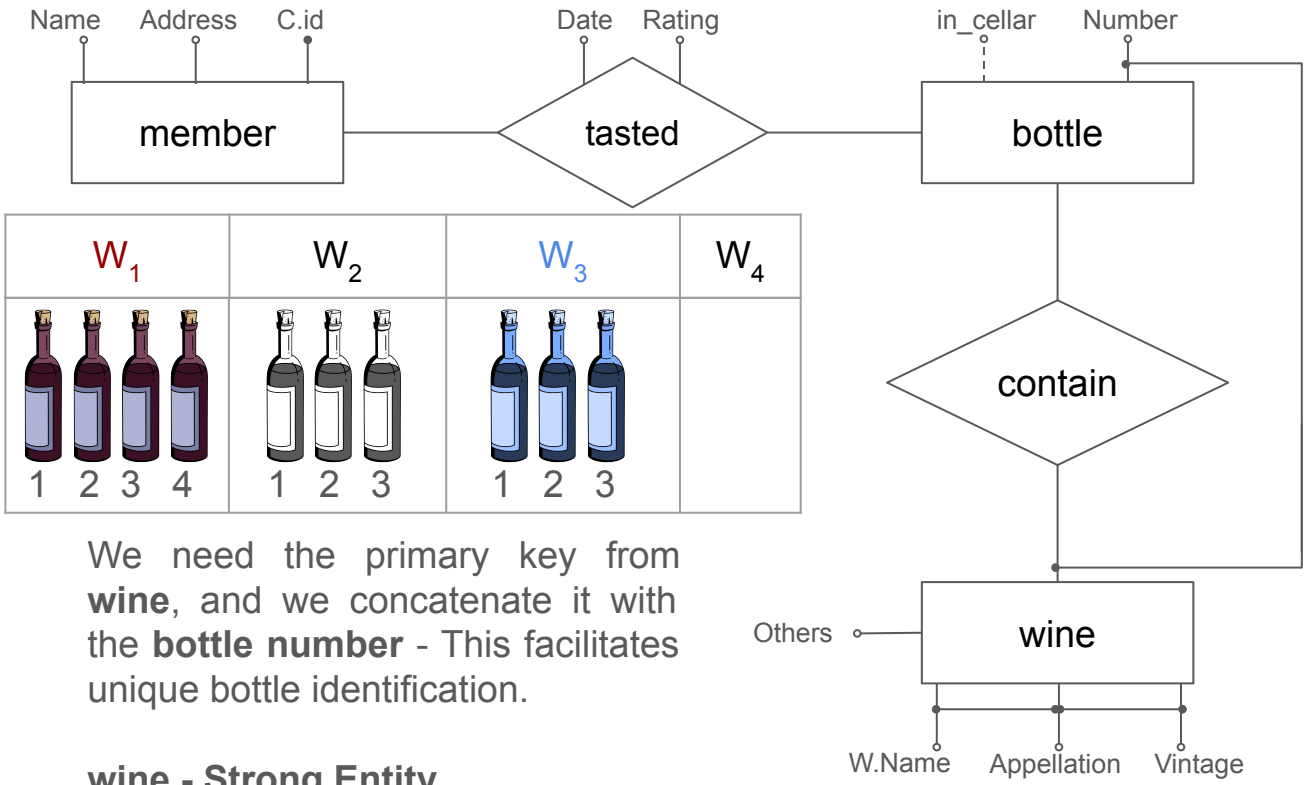


But, mentioning **Bottle 1** of **Wine 1 (W_1)** uniquely identifies a bottle.

Thus we need to identify the wine first, then the bottle - We need the primary key from **wine**, and we concatenate it with the **bottle number** - This facilitates unique bottle identification.

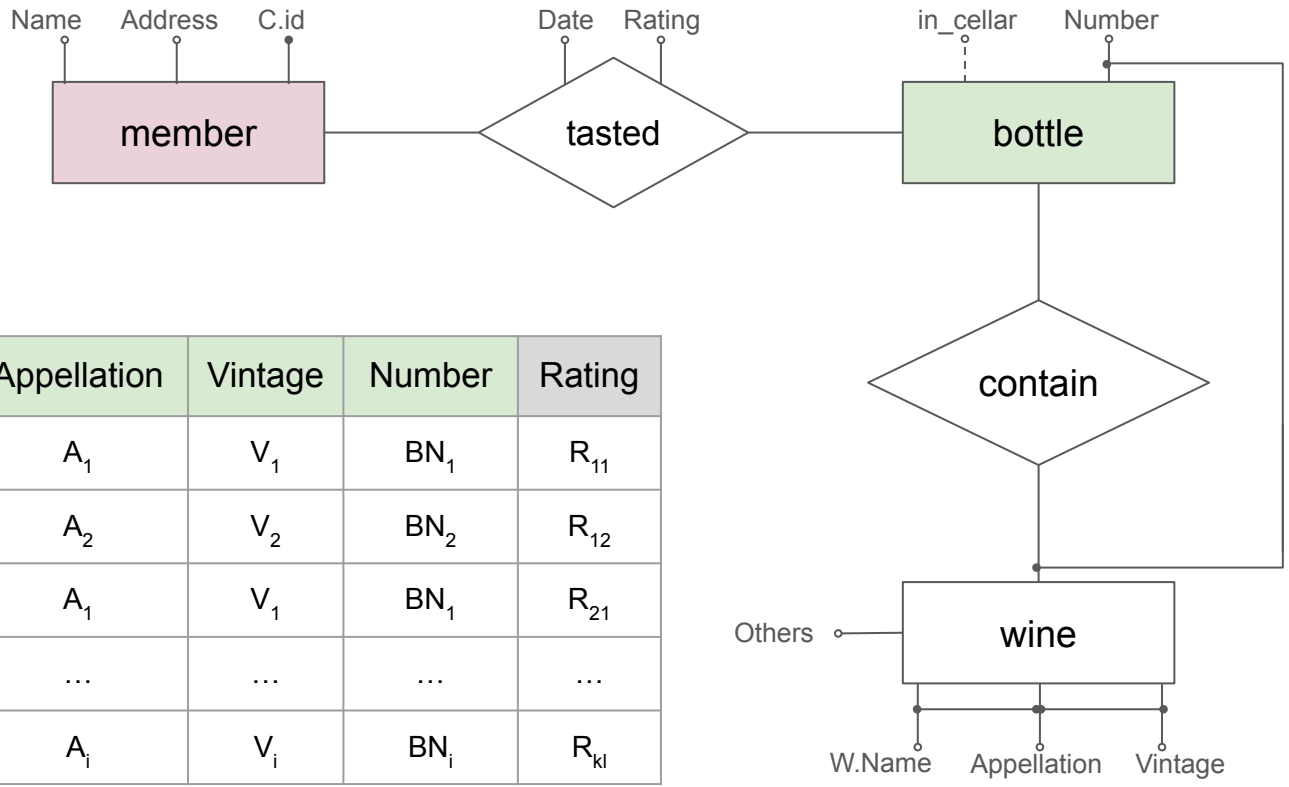
"Generally, there are or have been **several bottles of the same wine** in the cellar. **For each wine, the bottles in the wine cellar of VINO are numbered.** For instance, the cellar has 20 bottles numbered 1 to 20 of a Semillon from 1996 named Rumbalara. **For documentation purposes, VINO may also want to record wines for which it does not own bottles.** The bottles are **either available in the cellar or they have been tasted and emptied**."

Bottles have **Number** and **Status** as its own attributes.
But these are not enough to identify the bottles **UNIQUELY**.



We need the primary key from **wine**, and we concatenate it with the **bottle number** - This facilitates unique bottle identification.

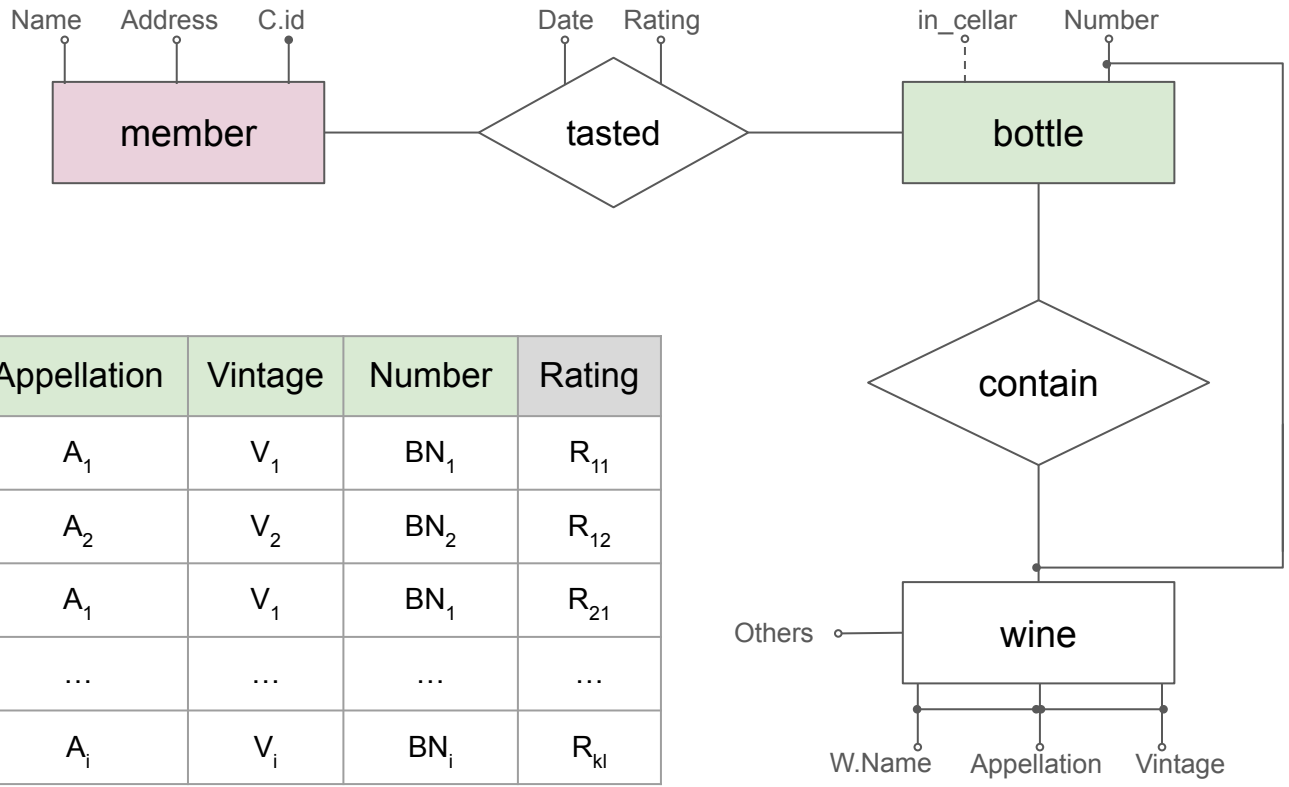
wine - Strong Entity
bottle - Weak Entity
Existence of a bottle of wine depends on the existence of the wine.



Date	C.id	Name	Appellation	Vintage	Number	Rating
D_1	C_1	N_1	A_1	V_1	BN_1	R_{11}
D_2	C_1	N_2	A_2	V_2	BN_2	R_{12}
D_3	C_2	N_1	A_1	V_1	BN_1	R_{21}
…	…	…	…	…	…	…
D_k	C_j	N_i	A_i	V_i	BN_i	R_{kl}

Example instantiation of the **tasted** relationship

"At each session, **the attending members** taste several bottles". - **Suggests that there can be non-attending members.**

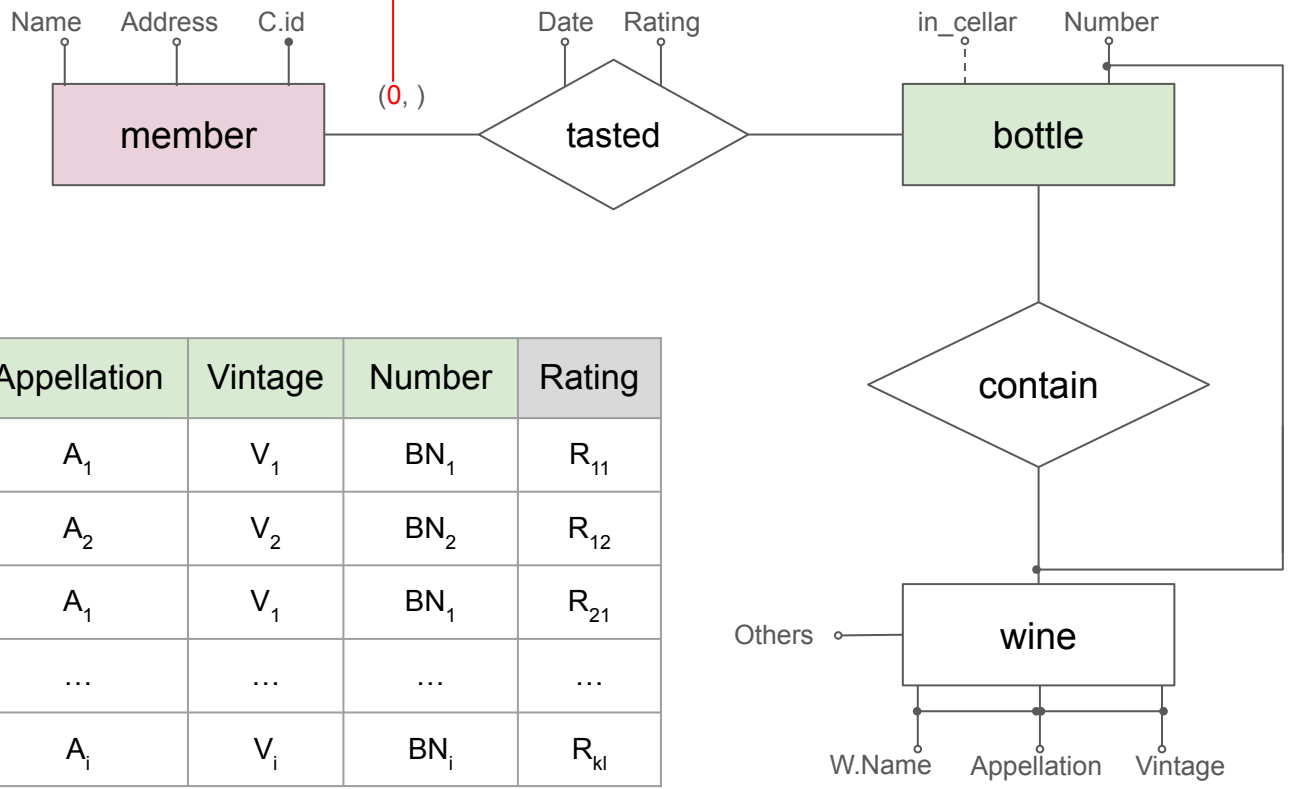


Date	C.id	Name	Appellation	Vintage	Number	Rating
D_1	C_1	N_1	A_1	V_1	BN_1	R_{11}
D_2	C_1	N_2	A_2	V_2	BN_2	R_{12}
D_3	C_2	N_1	A_1	V_1	BN_1	R_{21}
…	…	…	…	…	…	…
D_k	C_j	N_i	A_i	V_i	BN_i	R_{kl}

Example instantiation of the **tasted** relationship

There can be members who have **NOT** attended any session, thus have **NOT** tasted any bottle - Thus **DO NOT** appear in **Taste** relationship - Min constraint = 0

"At each session, **the attending members** taste several bottles". - **Suggests that there can be non-attending members.**

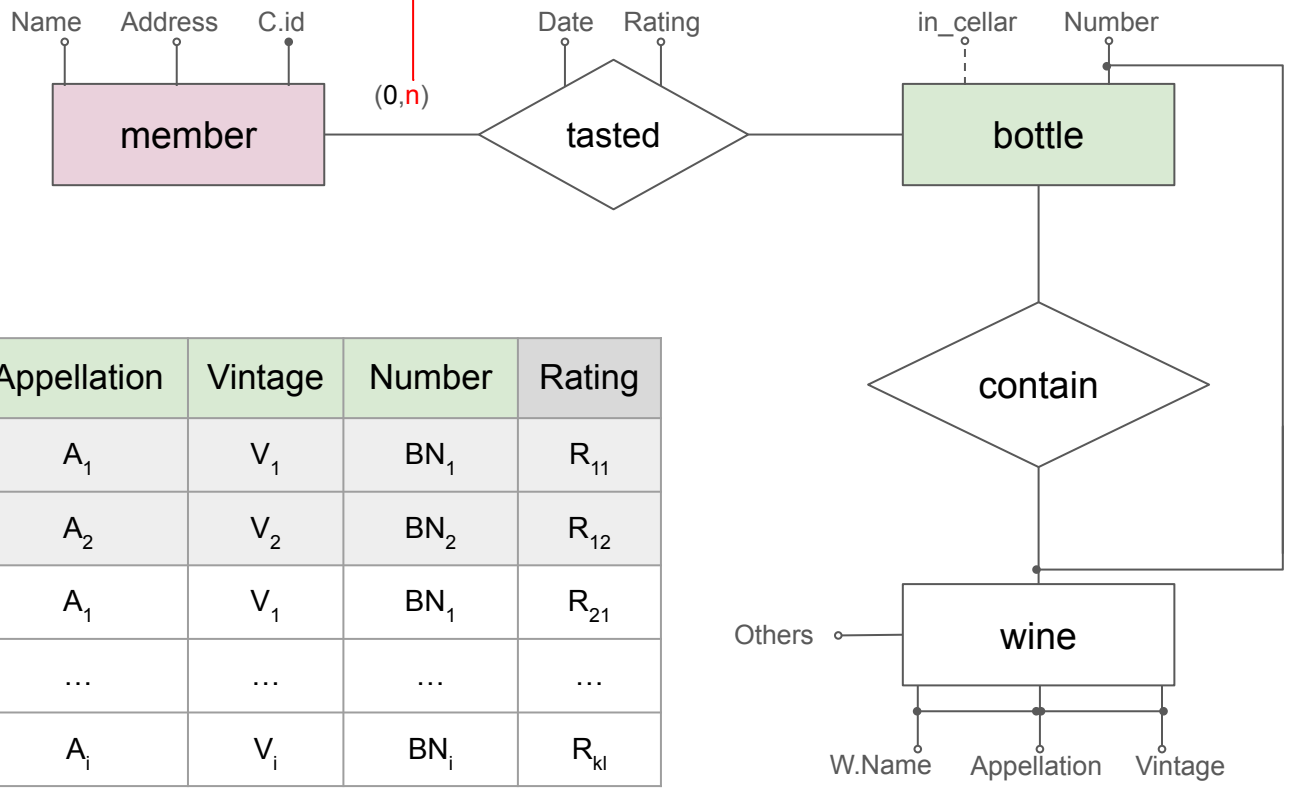


Date	C.id	Name	Appellation	Vintage	Number	Rating
D_1	C_1	N_1	A_1	V_1	BN_1	R_{11}
D_2	C_1	N_2	A_2	V_2	BN_2	R_{12}
D_3	C_2	N_1	A_1	V_1	BN_1	R_{21}
…	…	…	…	…	…	…
D_k	C_j	N_i	A_i	V_i	BN_i	R_{kl}

Example instantiation of the **tasted** relationship

There can be members who have tasted more than one bottles - Thus can appear in **Taste** relationship **MORE THAN ONCE** - Max constraint = n (denotes **MANY**)

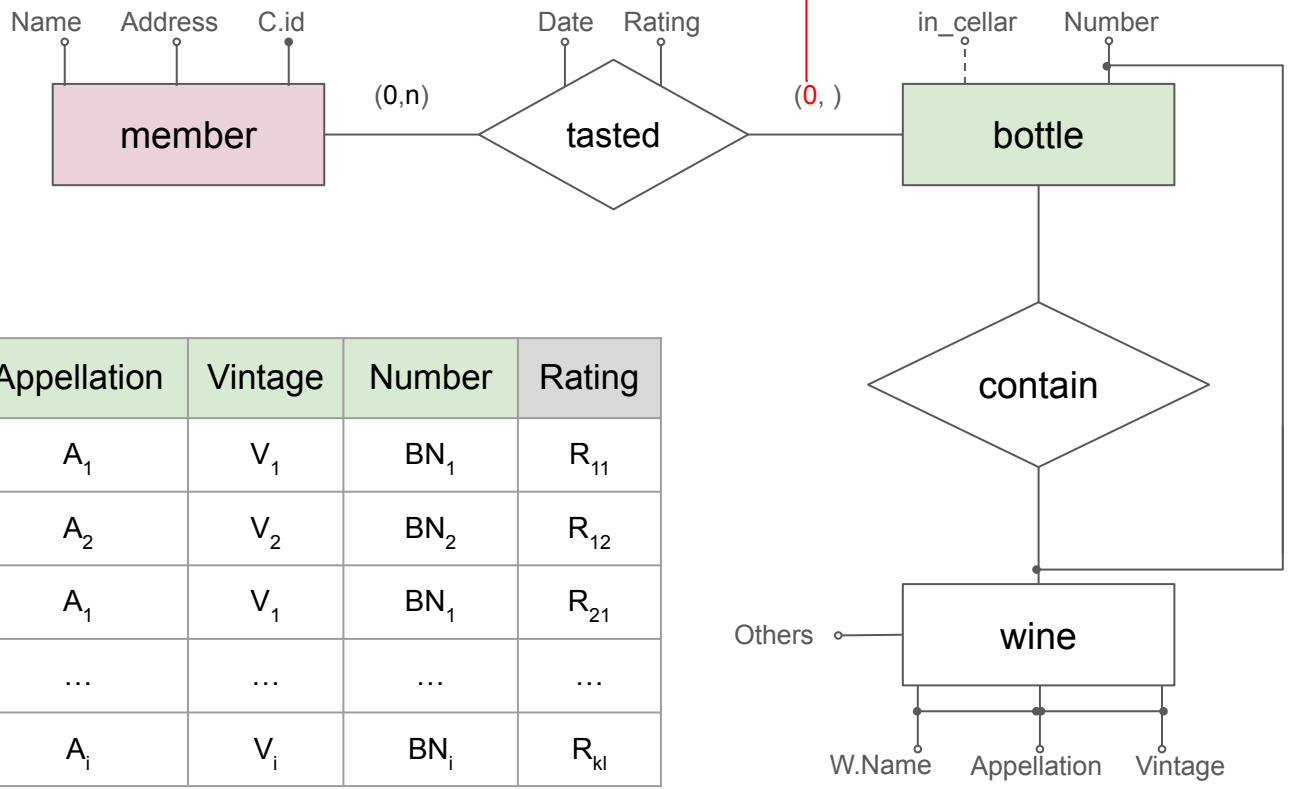
"At each session, the attending members **taste several bottles**." - **Suggests that there can be non-attending members.**



Date	C.id	Name	Appellation	Vintage	Number	Rating
D_1	C_1	N_1	A_1	V_1	BN_1	R_{11}
D_2	C_1	N_2	A_2	V_2	BN_2	R_{12}
D_3	C_2	N_1	A_1	V_1	BN_1	R_{21}
…	…	…	…	…	…	…
D_k	C_j	N_i	A_i	V_i	BN_i	R_{kl}

Example instantiation of the **tasted** relationship

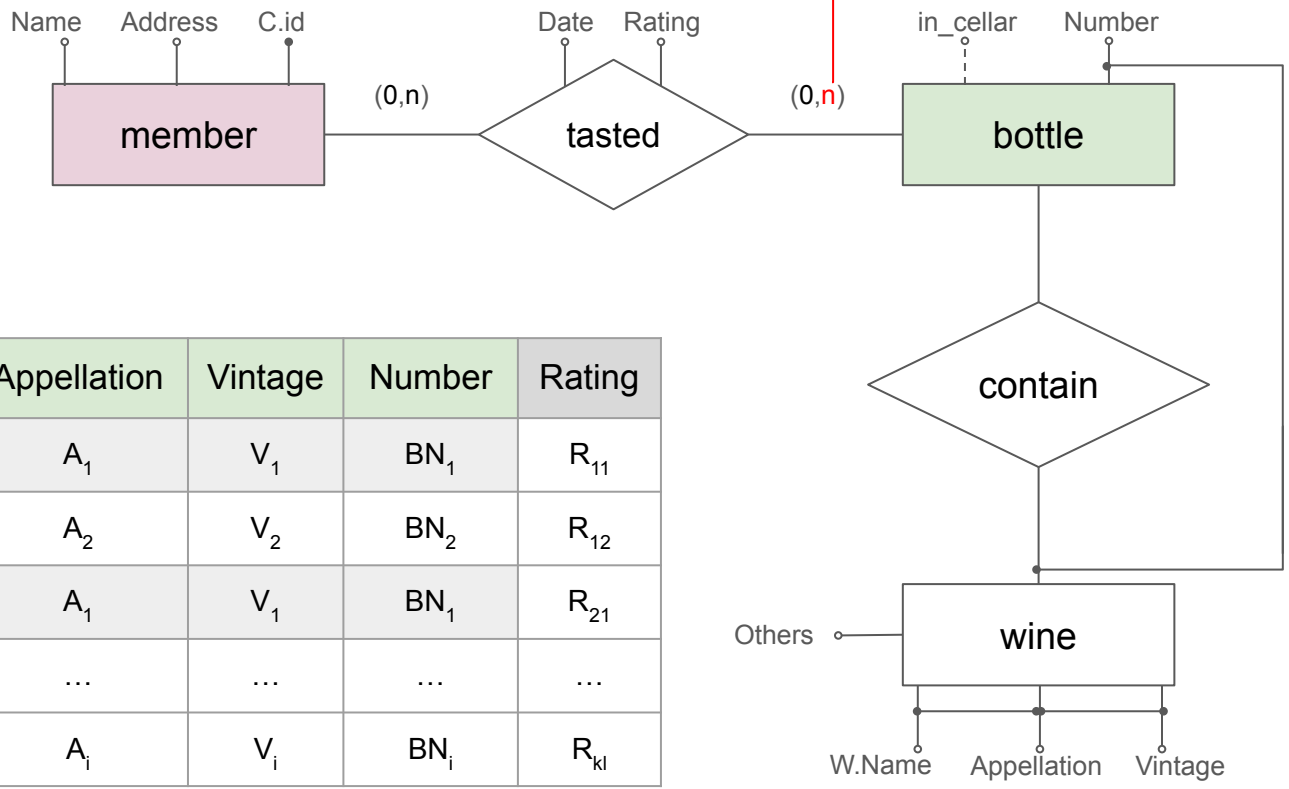
There can be bottles which **no member has tasted** - Thus they **DO NOT** appear in **Taste** relationship - Min constraint = 0



Date	C.id	Name	Appellation	Vintage	Number	Rating
D_1	C_1	N_1	A_1	V_1	BN_1	R_{11}
D_2	C_1	N_2	A_2	V_2	BN_2	R_{12}
D_3	C_2	N_1	A_1	V_1	BN_1	R_{21}
…	…	…	…	…	…	…
D_k	C_j	N_i	A_i	V_i	BN_i	R_{kl}

Example instantiation of the **tasted** relationship

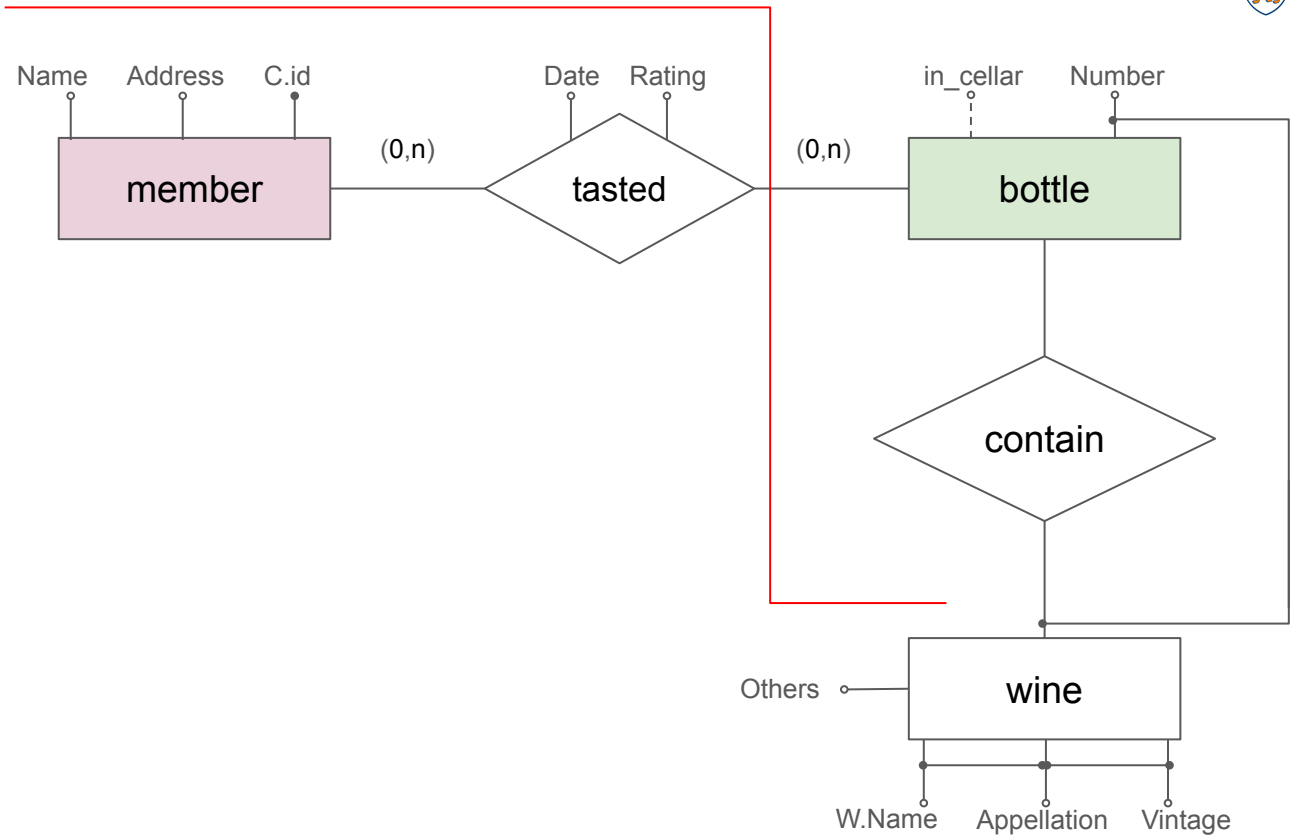
There can be bottles which **MANY members have tasted** - Thus they appear **MORE THAN ONCE** in **Taste** relationship - Max constraint = n (denotes **MANY**)



Date	C.id	Name	Appellation	Vintage	Number	Rating
D_1	C_1	N_1	A_1	V_1	BN_1	R_{11}
D_2	C_1	N_2	A_2	V_2	BN_2	R_{12}
D_3	C_2	N_1	A_1	V_1	BN_1	R_{21}
…	…	…	…	…	…	…
D_k	C_j	N_i	A_i	V_i	BN_i	R_{kl}

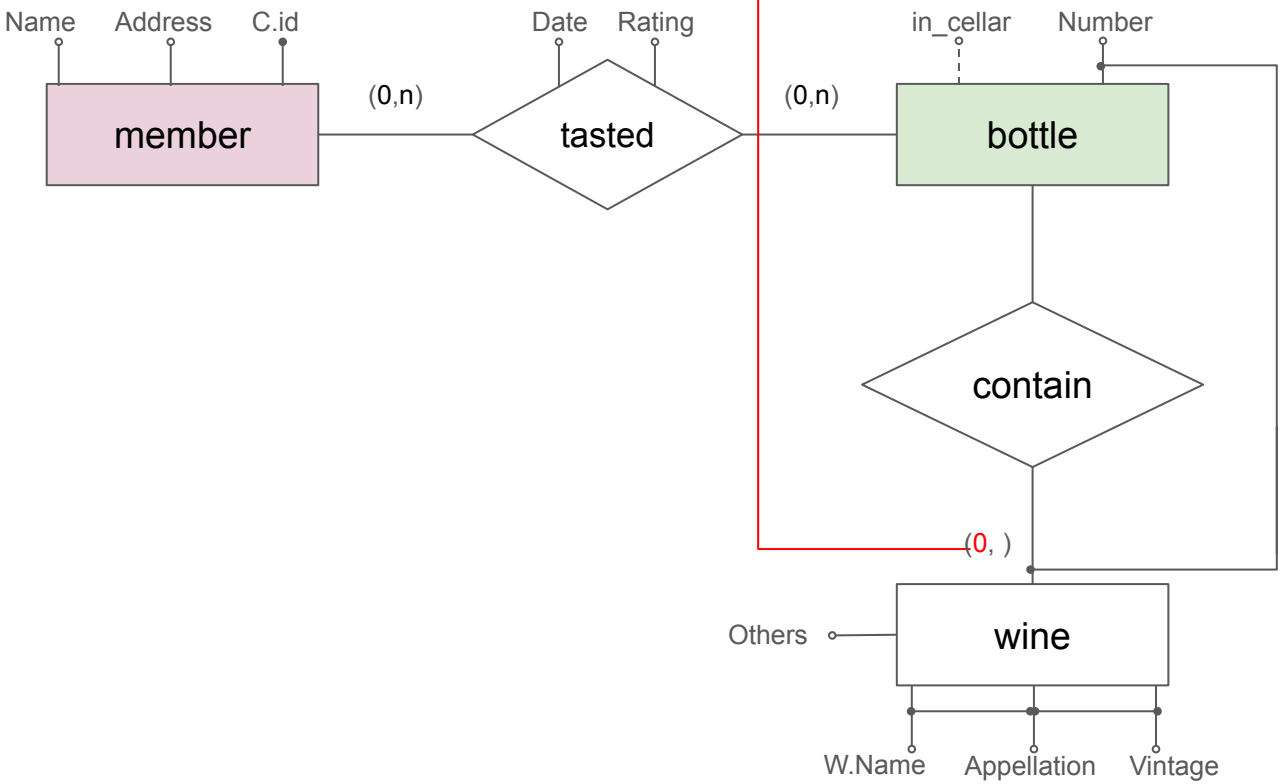
Example instantiation of the **tasted** relationship

“For documentation purposes, VINO may also want to **record wines for which it does not own bottles.**”



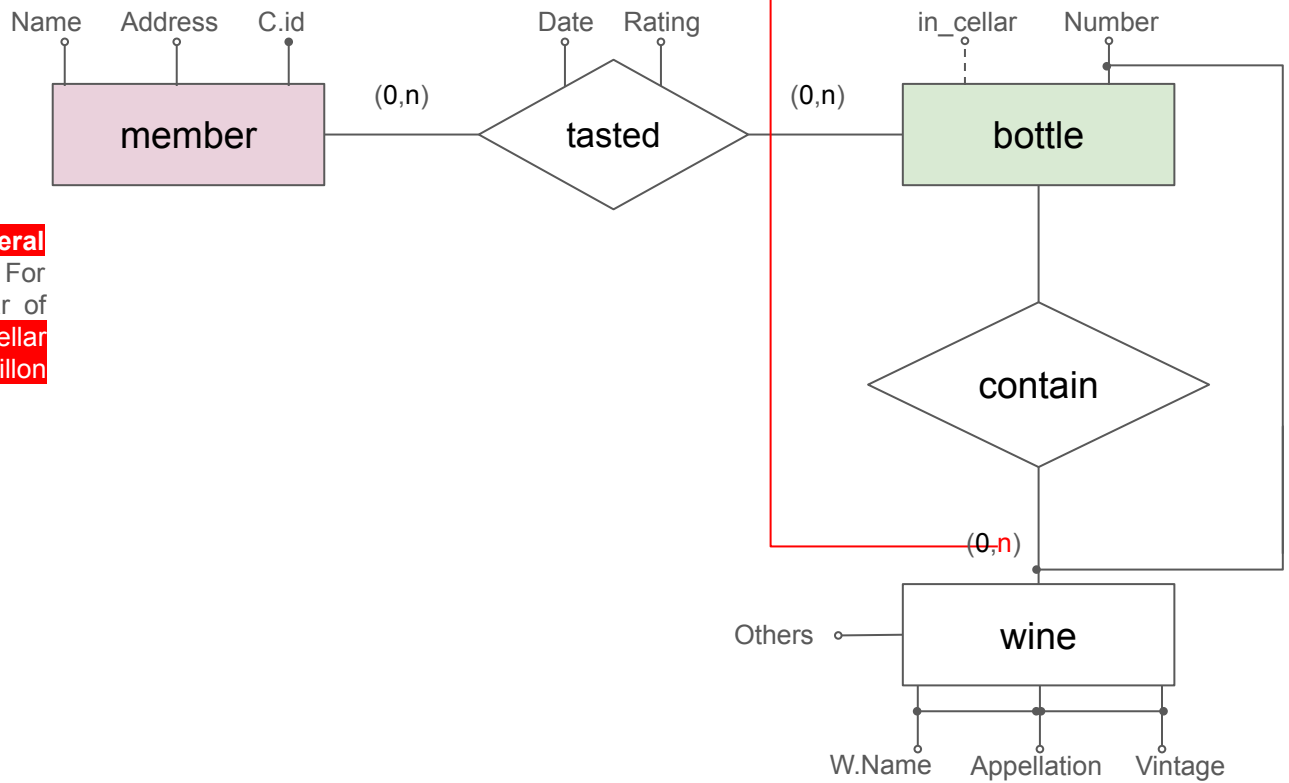
There can be wines, for which VINO has/had **no bottles** - Thus **DO NOT** appear in Contain relationship - Min constraint = 0

"For documentation purposes, VINO may also want to **record wines for which it does not own bottles.**"

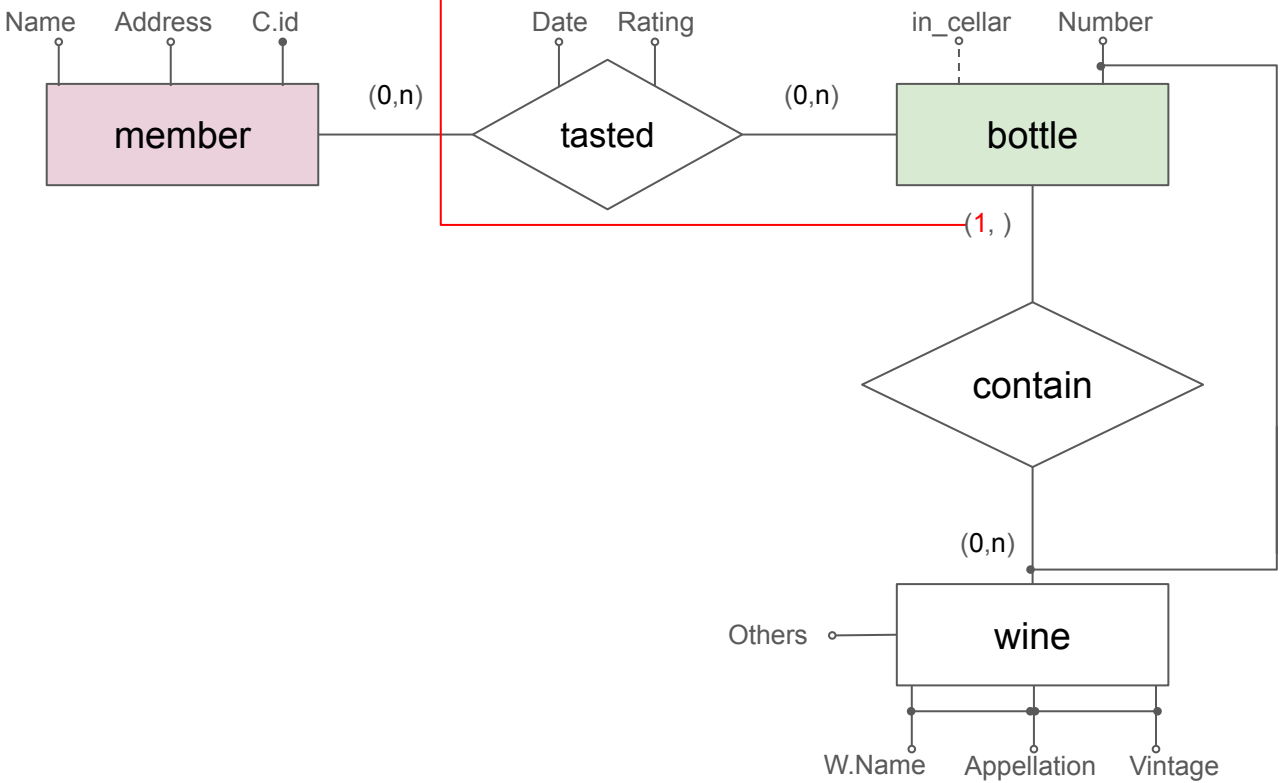


There can be **MANY** bottles of a wine - Thus can appear in Contain relationship **MORE THAN ONCE** - Max constraint = n(denotes **MANY**)

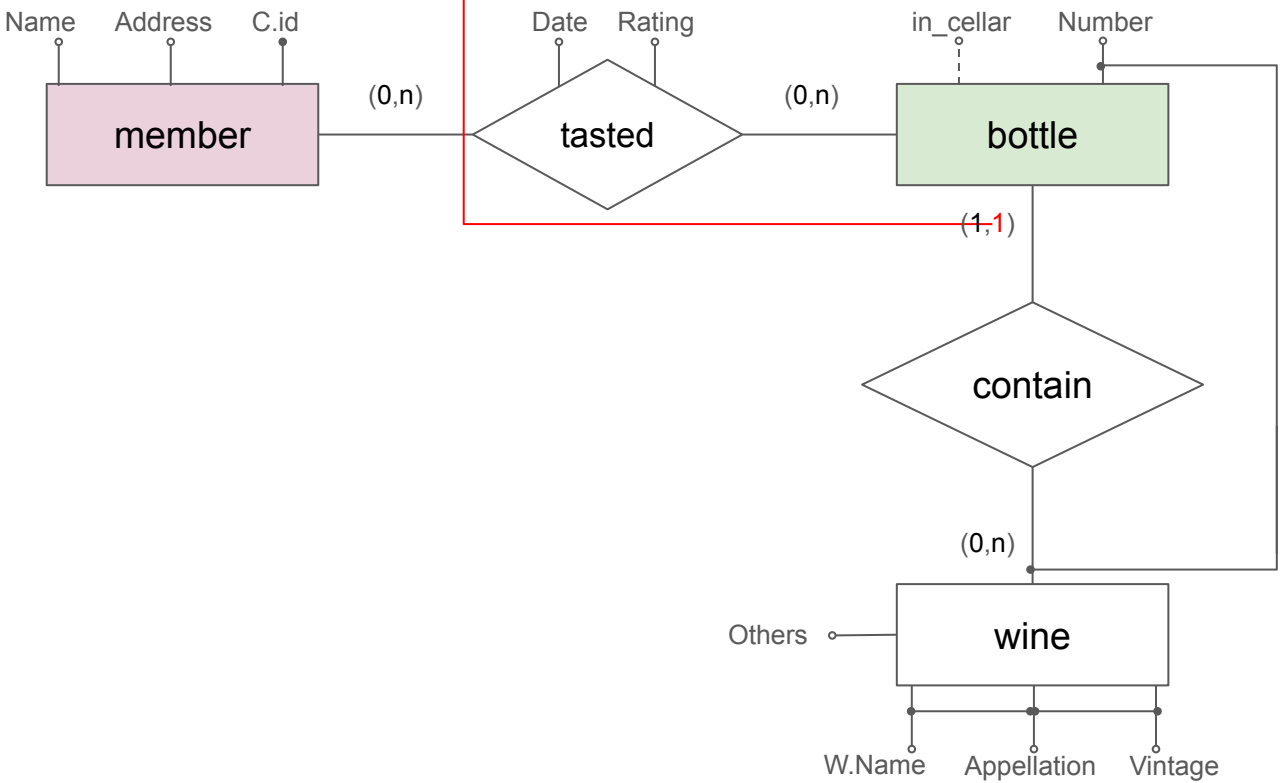
Generally, there **are or have been several bottles of the same wine in the cellar**. For each wine, the bottles in the wine cellar of VINO are numbered. For instance, the cellar has 20 bottles numbered 1 to 20 of a Semillon from 1996 named Rumbalara.



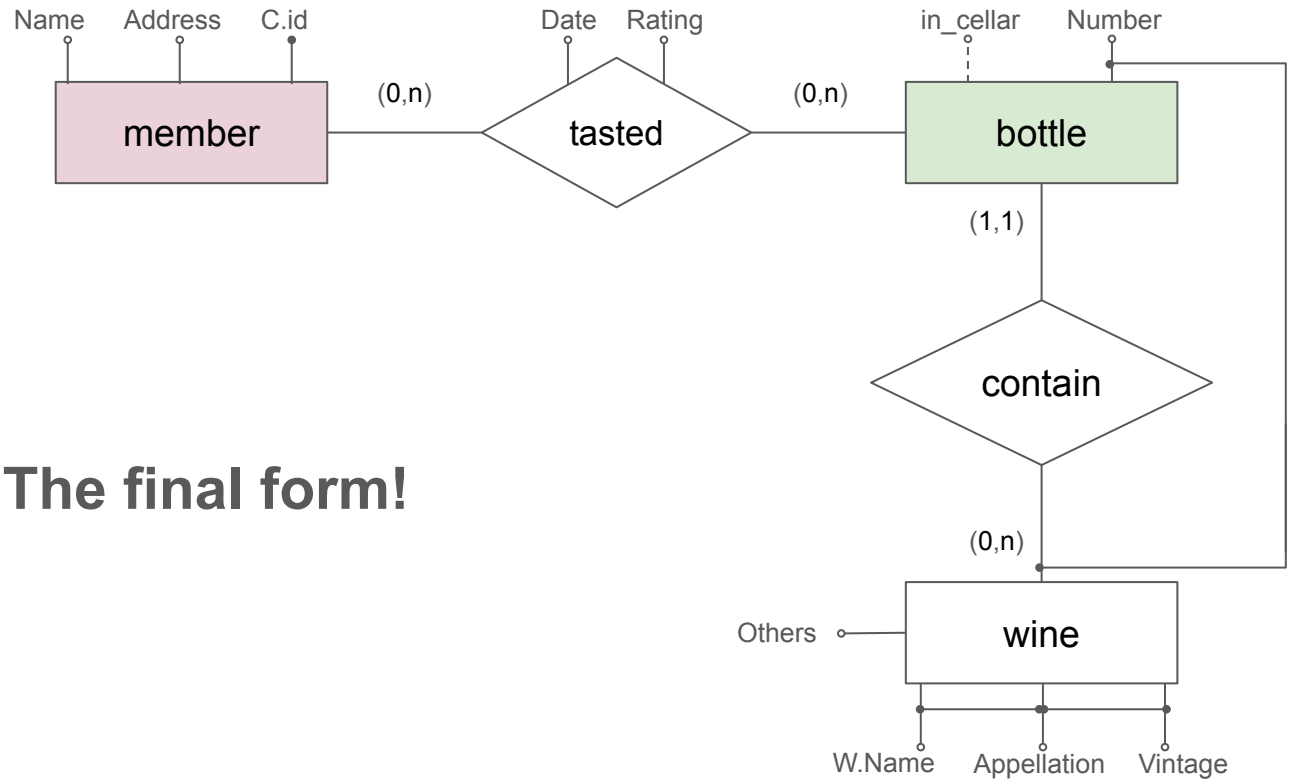
If there exists a bottle in the database, it has/had some wine - Thus it has to be in the Contain relationship - Min constraint = 1

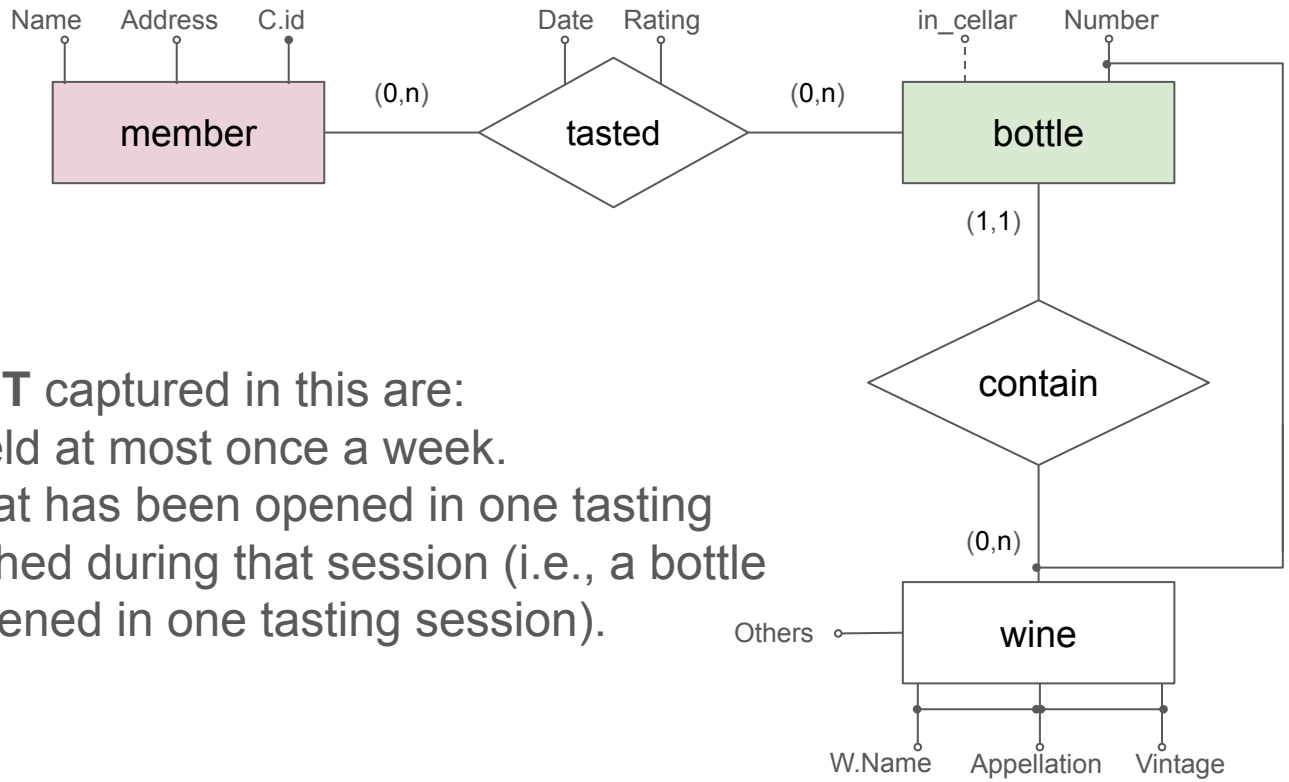


One bottle cannot contain more than one wines - Thus can appear in the Contain relationship only once - Max constraint = 1



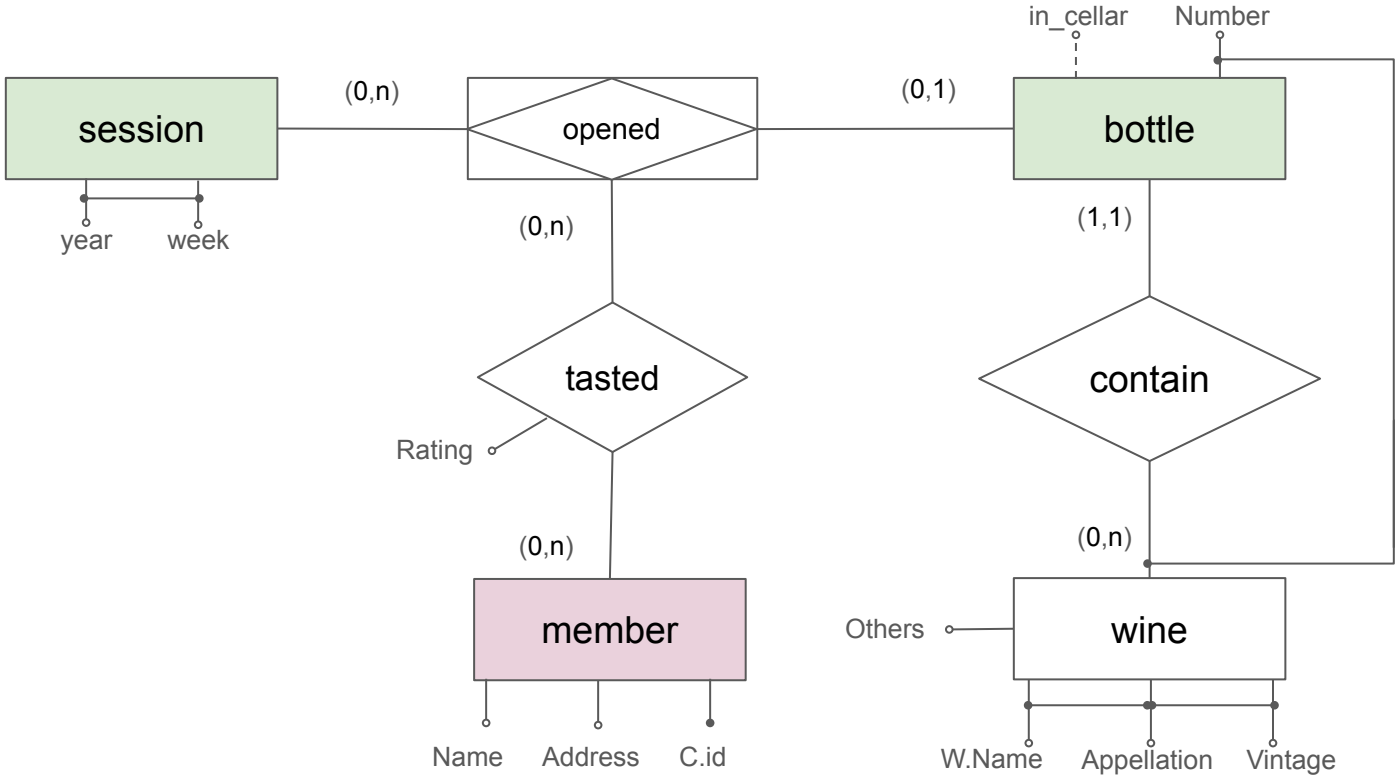
The final form!

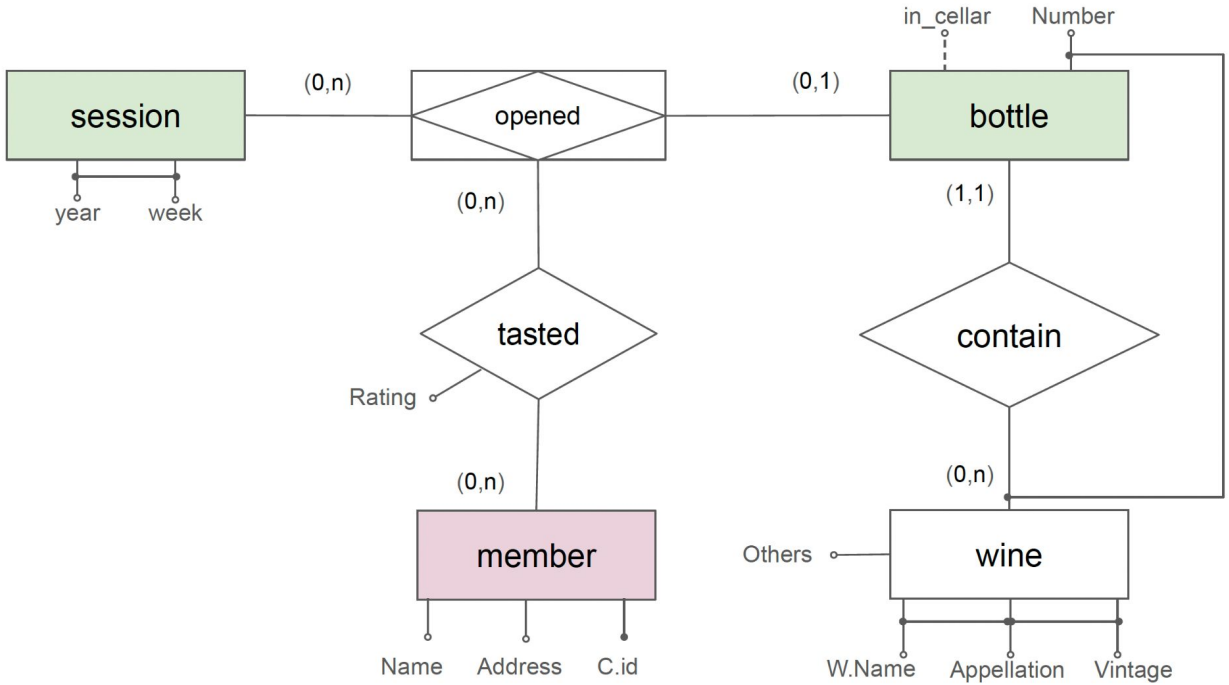




The constraints **NOT** captured in this are:

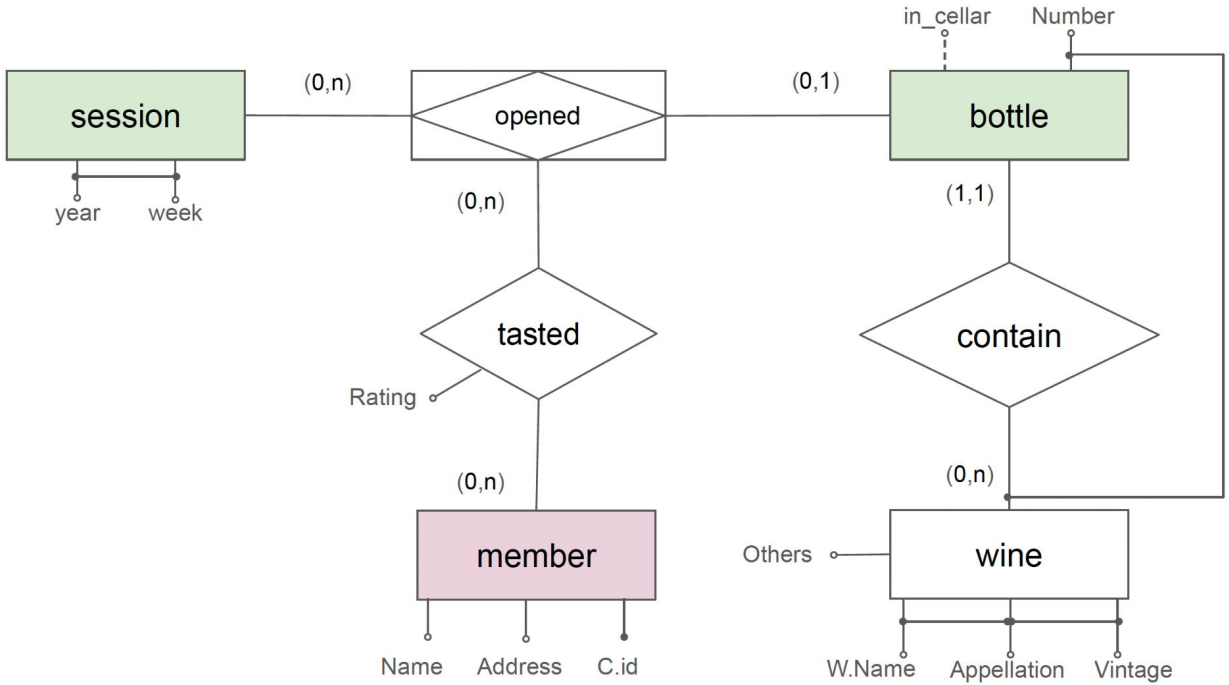
- A session is held at most once a week.
- Every bottle that has been opened in one tasting session is finished during that session (i.e., a bottle can only be opened in one tasting session).


session
(0,n)
opened
(0,1)
bottle
in_cellar
Number
year
week
(0,n)
(1,1)
tasted
contain
Rating
(0,n)
(0,n)
member
wine
Others
Name
Address
C.id
W.Name
Appellation
Vintage


session
year
week
(0,n)
opened
(0,1)
in_cellar
Number
bottle
(0,n)
(1,1)
tasted
Rating
contain
(0,n)
(0,n)
member
Name
Address
C.id
Others
wine
W.Name
Appellation
Vintage

📌 NOTE:

Aggregation = treat a relationship (with its participating entities) as a single higher-level unit so another relationship can attach to that unit.



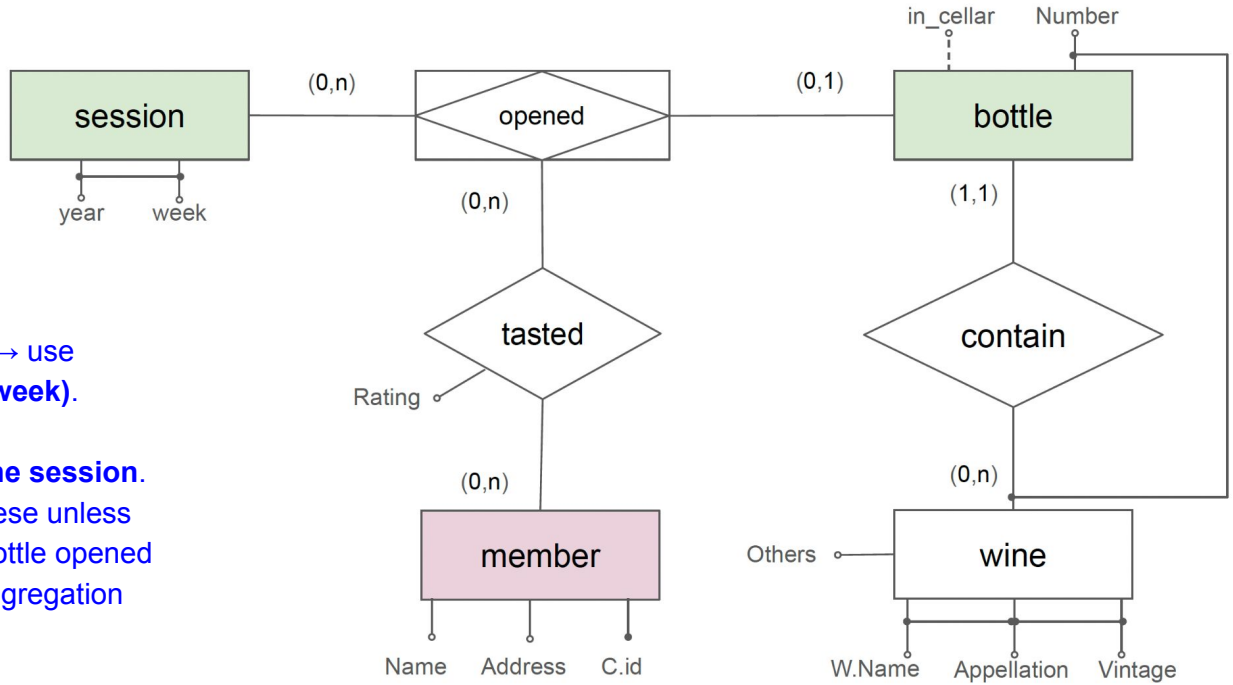
📌 NOTE:

Aggregation = treat a relationship (with its participating entities) as a single higher-level unit so another relationship can attach to that unit.

🤔 Why we need it in VINO?

We must model two rules precisely:

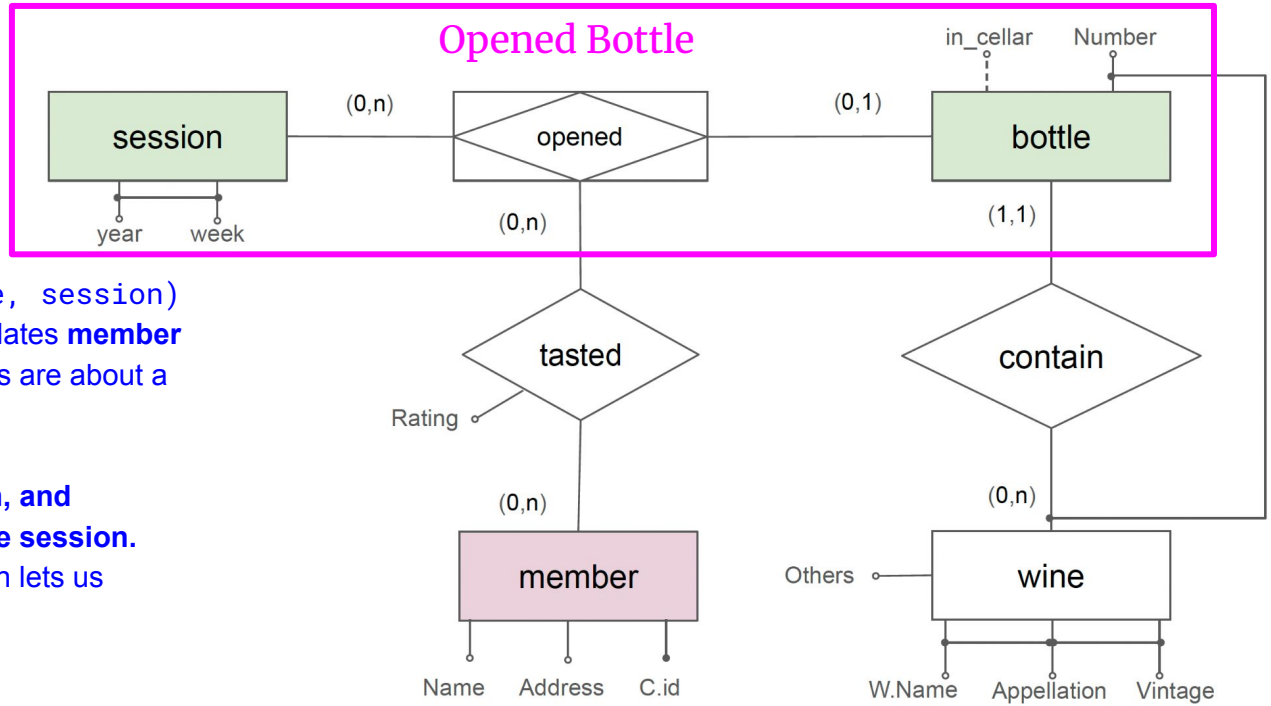
- **At most one session per week** → use `session` with natural key **(year, week)**.
- **A bottle is opened in at most one session**. Plain ER can't cleanly enforce these unless we attach tastings to the *event* "bottle opened in session," not to a raw bottle. Aggregation gives us that attachment point.



📌 NOTE:

Aggregation = treat a relationship (with its participating entities) as a single higher-level unit so another relationship can attach to that unit.

🤔 What it's doing in the diagram?



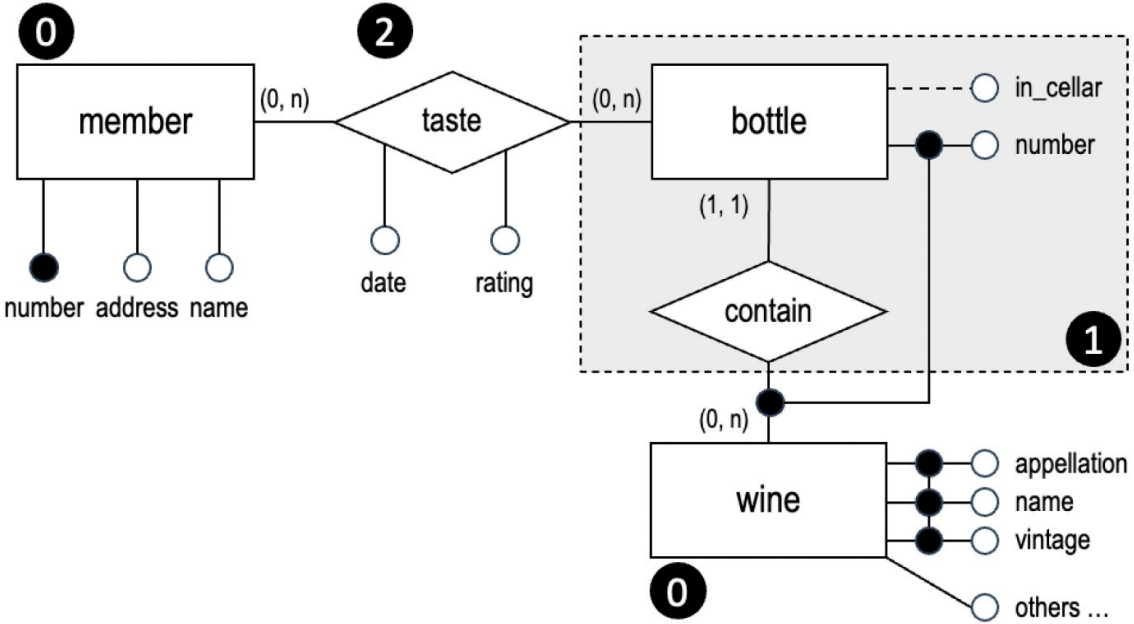
- We **package** `opened(bottle, session)` as a single unit; then **tasted** relates **member** → **[opened-bottle]** (i.e., ratings are about a bottle *in its session*).
- **Bottle is opened in a session, and members taste it in that same session.** That's exactly what aggregation lets us model—ratings attach to *that* opened-bottle-in-that-session.
- **Models the event:** a **bottle** is **opened** in a specific **session**, and **members taste it there**. Ratings are about that event, not the abstract bottle.

2(a). Translate ER diagram to relationship schema

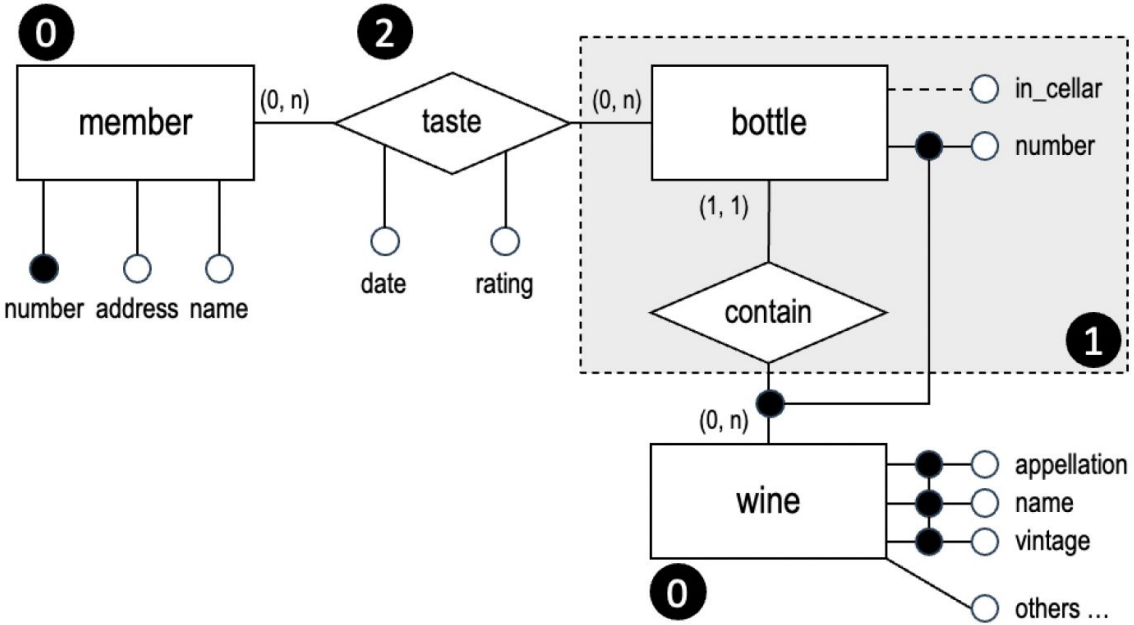
Rule of thumb:

- **Create any table with 0 unresolved foreign keys.**
- **Merge** a weak entity with its **identifying** relationship *(when the relationship has no attributes)*; set **PK = owner PK + partial key**, and keep **FK → owner**.
- **Recount unresolved FKs and repeat** until nothing remains.

2(a). Translate ER diagram to relationship schema



2(a). Translate ER diagram to relationship schema



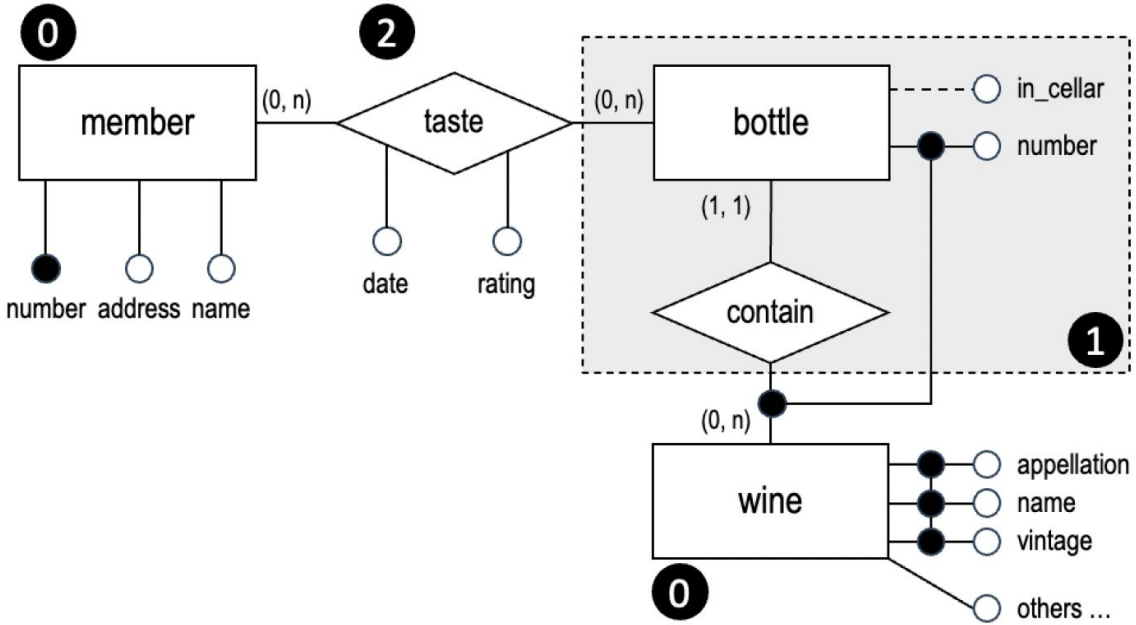
Entities/relationships in the diagram:
`member`, `wine`, `bottle`(weak) + `contain`(identifying), `taste`.

FK count before starting:

- `member`: **0** (strong entity)
- `wine`: **0** (strong entity)
- `bottle+contain` (merged): **1** → needs FK to `wine`
- `taste`: **2** → needs FK to `member` and to `bottle`

Process = "translate everything with count **0**; update counts; repeat."

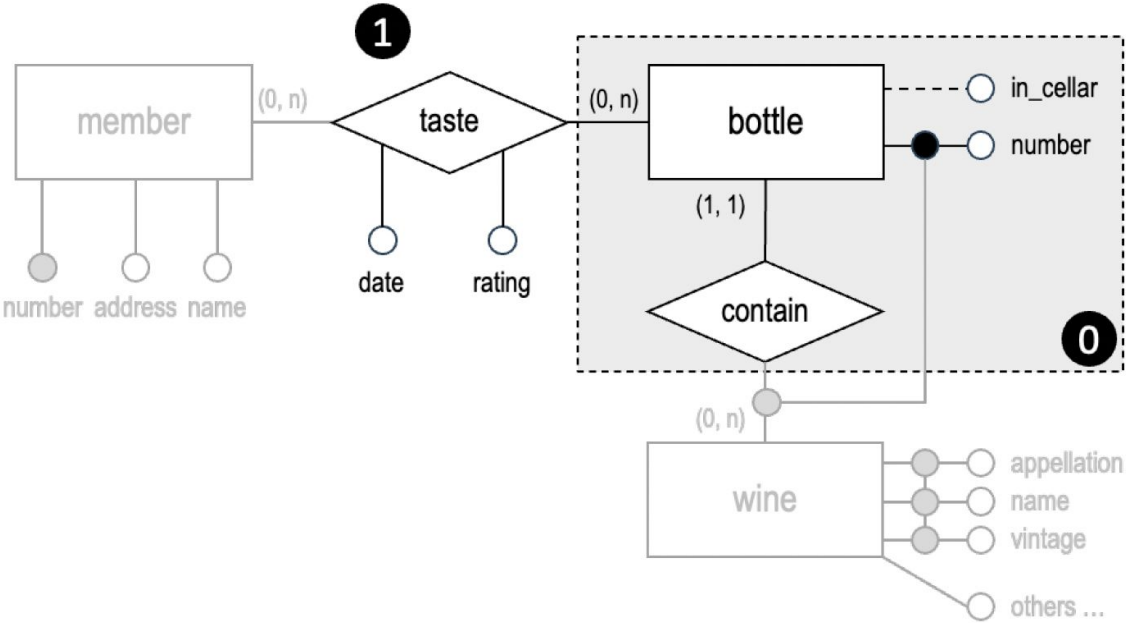
Step 1: Translate member and wine




```
CREATE TABLE member (
  card_number  CHAR(10)     PRIMARY KEY,
  address      VARCHAR(64)  NOT NULL,
  name         VARCHAR(32)  NOT NULL
);

CREATE TABLE wine (
  name             VARCHAR(32),
  appellation      VARCHAR(32),
  vintage          DATE,
  alcohol_degree   NUMERIC NOT NULL,
  bottled          VARCHAR(128) NOT NULL,
  certification    VARCHAR(64),
  country          VARCHAR(32) NOT NULL,
  PRIMARY KEY (name, appellation, vintage)
);
```

Step 2: Translate **bottle + contain** (merged)

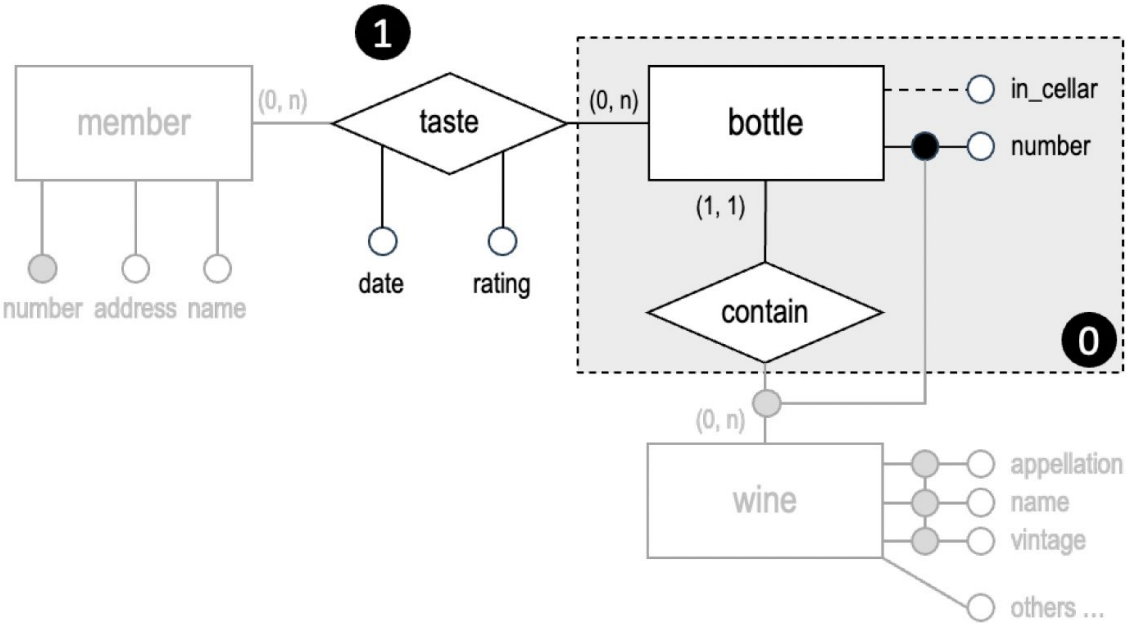


🤔When can we merge a relationship into an entity's table?

If a weak entity's identifying relationship has no attributes, merge the weak entity with that relationship into a single table.

Why safe: There's no information lost—the relationship contributes no attributes of its own.

Step 2: Translate **bottle + contain** (merged)



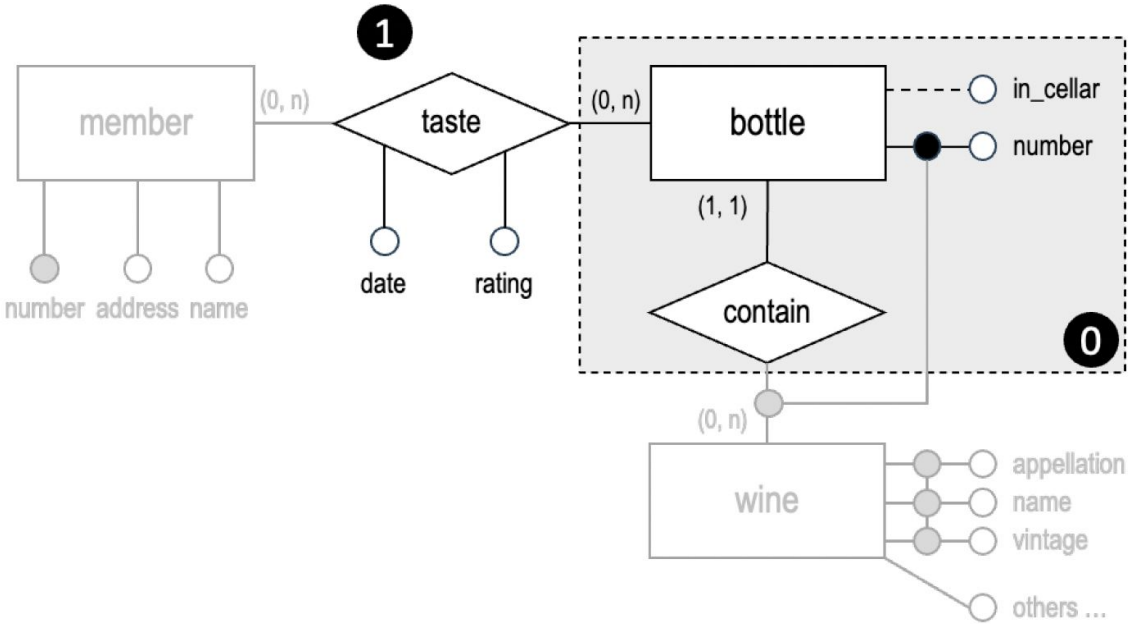
🤔Why now?

💡Its only dependency (FK) is to wine, which exists after Step 1.

🤔Why merge?

💡`contain` is the identifying relationship for weak entity `bottle` and has no own attributes.

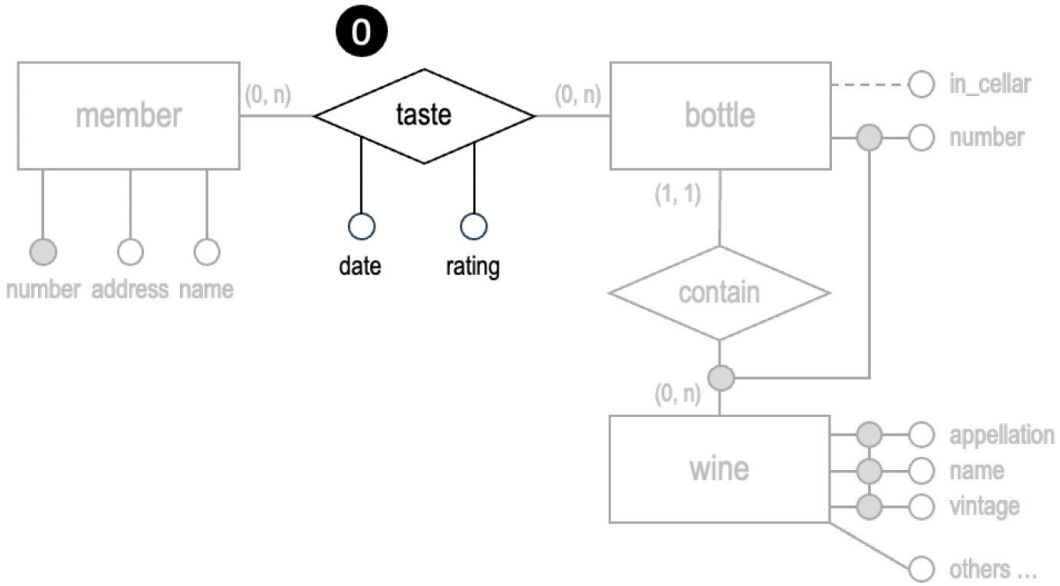
Step 2: Translate **bottle + contain** (merged)



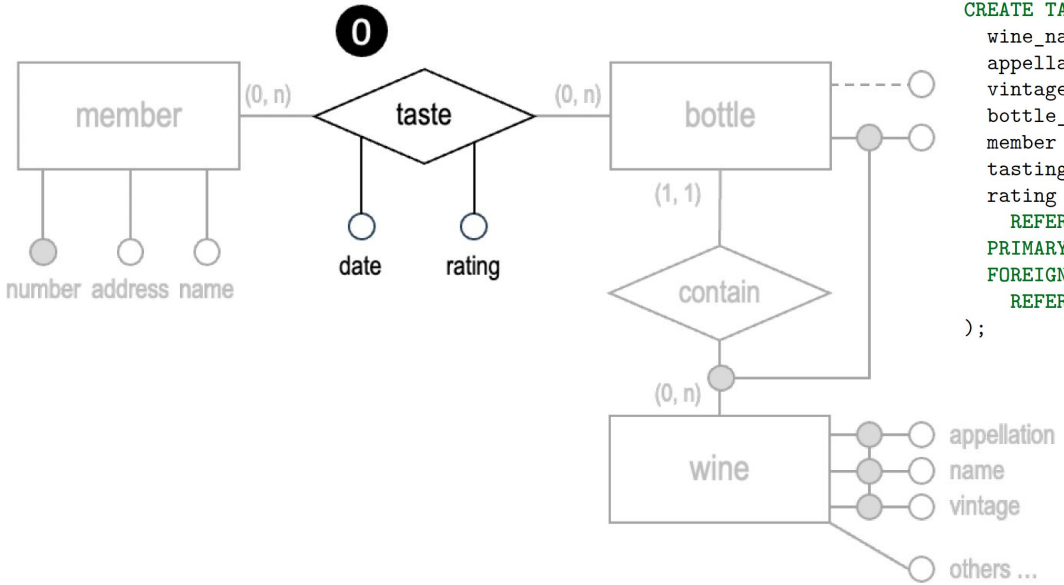

```
CREATE TABLE bottle (
  wine_name       VARCHAR(32),
  appellation     VARCHAR(32),
  vintage         DATE,
  number          INTEGER NOT NULL CHECK (number > 0),
  PRIMARY KEY (number, wine_name, appellation, vintage),
  FOREIGN KEY (wine_name, appellation, vintage)
    REFERENCES wine (name, appellation, vintage)
);
```

Implementation: Use the owner's key + weak entity's partial key as the **PRIMARY KEY**, and keep a **FOREIGN KEY** to the owner.

Step 3: Translate taste



Step 3: Translate taste




```
CREATE TABLE taste (
  wine_name      VARCHAR(32),
  appellation    VARCHAR(32),
  vintage        DATE,
  bottle_number  INTEGER,
  member         CHAR(10),
  tasting_date   DATE NOT NULL,
  rating         VARCHAR(32) NOT NULL
    REFERENCES member (card_number),
  PRIMARY KEY (member, bottle_number, wine_name, appellation, vintage),
  FOREIGN KEY (bottle_number, wine_name, appellation, vintage)
    REFERENCES bottle (bottle_number, wine_name, appellation, vintage)
);
```

Full Code

```
CREATE TABLE member (
  card_number CHAR(10) PRIMARY KEY,
  address     VARCHAR(64) NOT NULL,
  name        VARCHAR(32) NOT NULL
);

CREATE TABLE wine (
  name            VARCHAR(32),
  appellation     VARCHAR(32),
  vintage         DATE,
  alcohol_degree  NUMERIC NOT NULL,
  bottled         VARCHAR(128) NOT NULL,
  certification   VARCHAR(64),
  country         VARCHAR(32) NOT NULL,
  PRIMARY KEY (name, appellation, vintage)
);
```

```
CREATE TABLE bottle (
  wine_name    VARCHAR(32),
  appellation  VARCHAR(32),
  vintage      DATE,
  number       INTEGER NOT NULL CHECK
(number > 0),
  PRIMARY KEY (number, wine_name,
appellation, vintage),
  FOREIGN KEY (wine_name, appellation,
vintage)
    REFERENCES wine (name,
appellation, vintage)
);
```

Full Code

```
CREATE TABLE taste (
  wine_name      VARCHAR(32),
  appellation    VARCHAR(32),
  vintage        DATE,
  bottle_number  INTEGER,
  member         CHAR(10),
  tasting_date   DATE NOT NULL,
  rating         VARCHAR(32) NOT NULL,
  PRIMARY KEY (member, bottle_number, wine_name, appellation,
vintage),
  FOREIGN KEY (member)
    REFERENCES member (card_number),
  FOREIGN KEY (bottle_number, wine_name, appellation, vintage)
    REFERENCES bottle (number, wine_name, appellation, vintage)
);
```

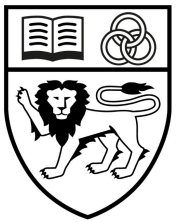
Thank you for joining!

Got questions? Post them on the forum or email me:

biswadeep@u.nus.edu

(I reply **within 2 working days** — *faster if coffee is strong* ☕)

Because your learning matters to me! 😊



NUS
National University
of Singapore