

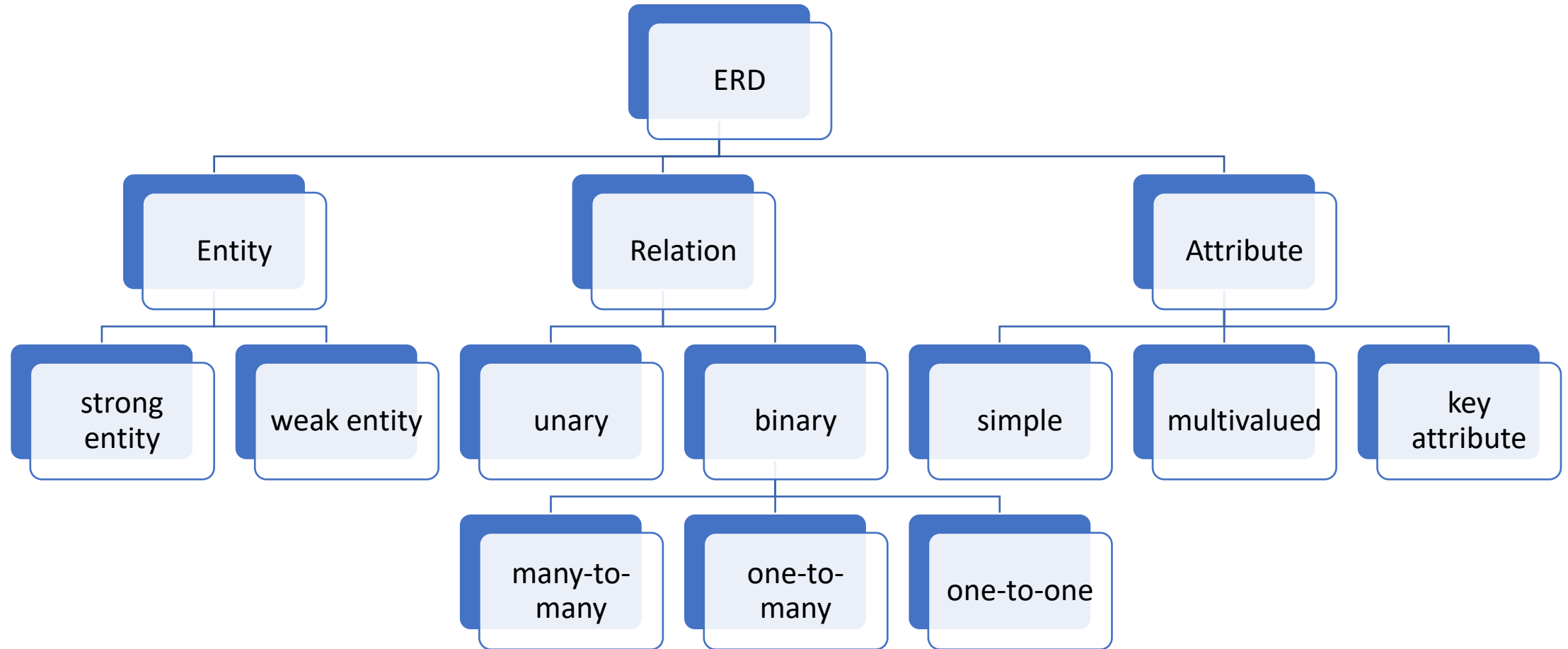
ER - DIAGRAM

LECTURE 2: Databases

ER - Diagram

- User story/requirement analysis ➔ **ER** ➔ relational database schema.
- Easy to translate into relational tables.
- High-level design.
- Suitable for structured systems.

ERD - components



ER - Diagram



person, place, activity, event, concept, real world object etc.
usually a noun



ER - Diagram



ENTITY

person, place, activity, event, concept, real world object etc.
usually a noun



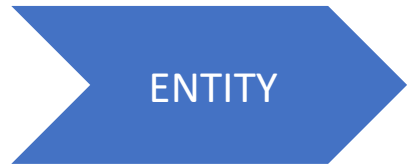
RELATION

links entities (unary, binary, ternary).
usually a verb



ATTRIBUTE

ER - Diagram



person, place, activity, event, concept, real world object etc.
usually a noun

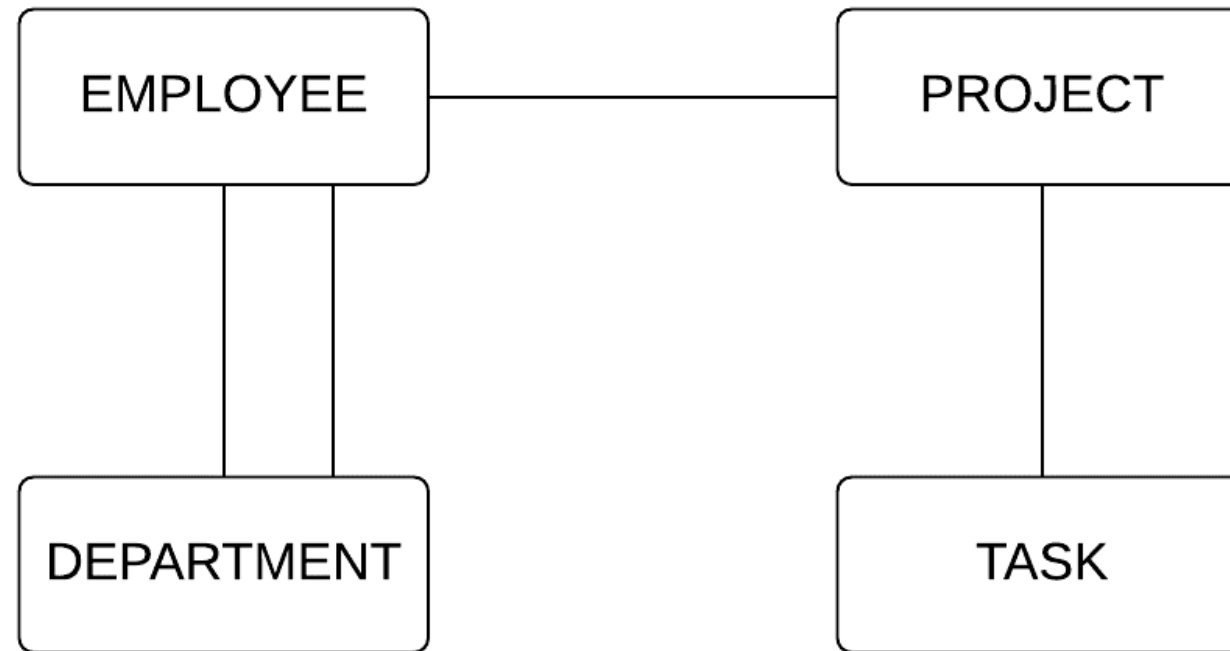


links entities (unary, binary, ternary).
usually a verb



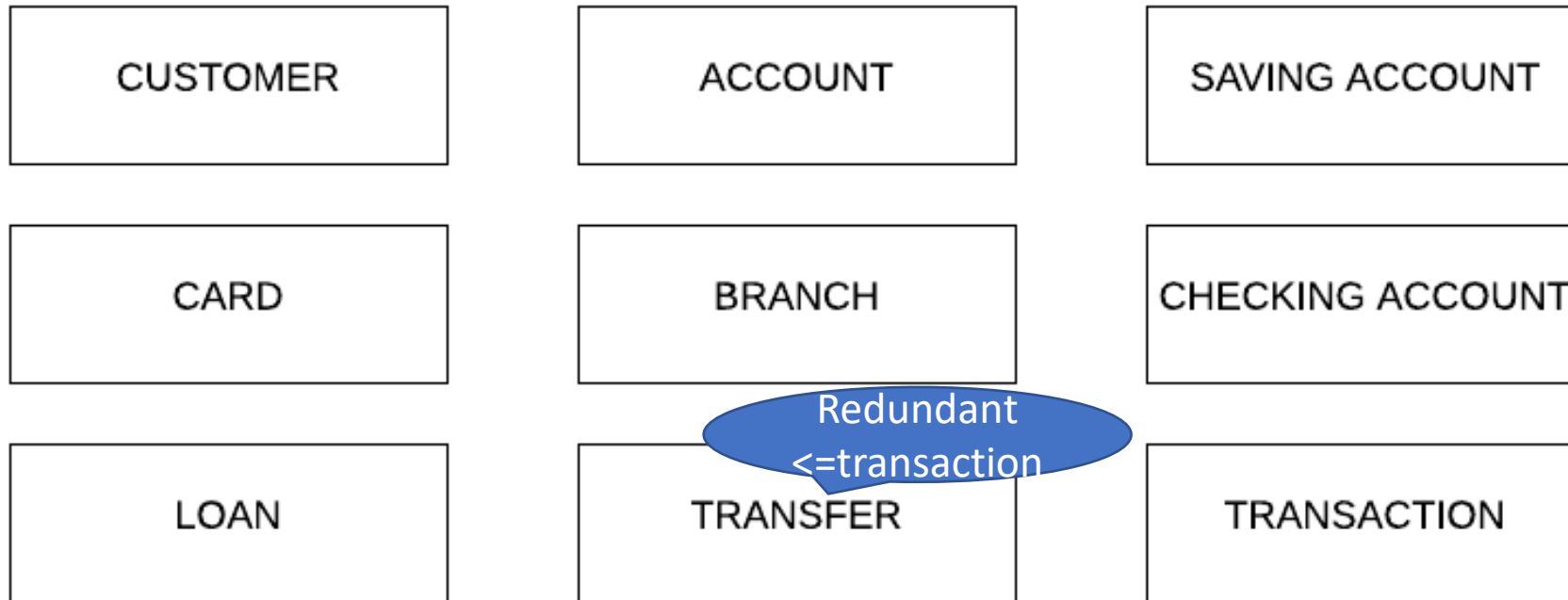
describe entities or relations

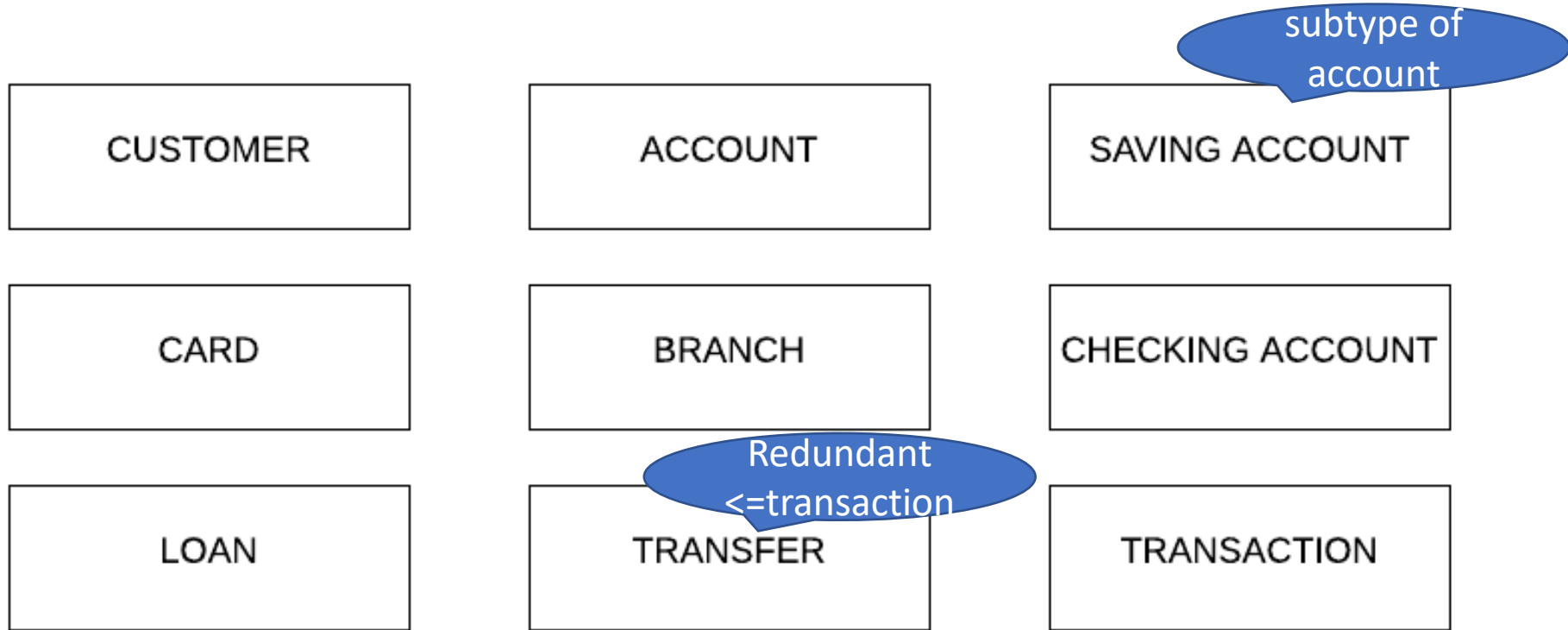
Entities

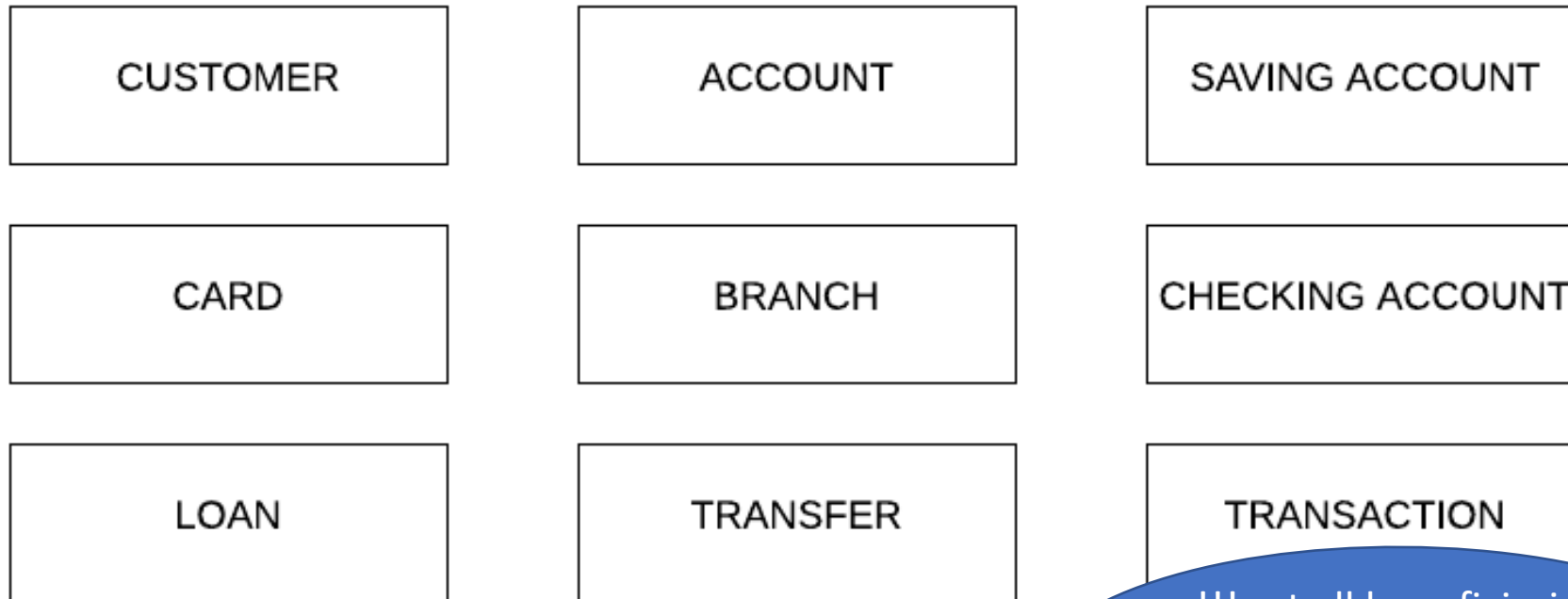


Banking (1)Entities

- A customer opens a saving account or a checking account, at a bank branch. He may also access loans. For each checking account he has a card. Periodically he may withdraw money from his account or partially pay his loans. He may also transfer money from one account to another.
- Please answer www.menti.com 8045 7859 Q5, Q6







!!!not all beneficiaries
(missing from story) are
customers of the same
bank

Entities

- Unique names, uppercase characters
- Graphical representation: rectangles
- Relational database: entity ➔ table (line & columns)
- Primary key: attribute or group of attributes that uniquely identifies entity instances

Candidate key

- *Relation* in relational model \leftrightarrow *relationship* in ERD
- Relation \leftrightarrow set of tuples \leftrightarrow tables \leftrightarrow attributes of relation are columns
- Minimal *superkey* or *o relation* (minimal set of attributes) such that:
 - 1) There are no two distinct tuples sharing the same values for the *superkey* (unique)
 - 2) No proper subsets of the superkey has property (1)
- Attributes of the superkey are called prime attributes.
- Attributes that does not occur in any superkey are non-prime attributes.
- If there are no null values, since a relation is a *set* of tuples, each relation has at least one candidate key, i.e the set of all its attributes.

Candidate keys

BUYER_ID	PRICE	HALL_NO	DATE	TYPE
1	150	Coliseum	08/03/22	VIP
1	150	Lyttelton	14/04/22	A
2	140	Olivier	01/05/22	A
2	90	Coliseum	04/06/22	B
2	220	Lyttelton	08/03/22	VIP
3	140	Olivier	14/04/22	B
3	220	Olivier	20/03/22	VIP

Candidate keys

BUYER_ID	PRICE	HALL_NO	DATE	TYPE
1	150	Coliseum	08/03/22	VIP
1	150	Lyttelton	14/04/22	A
2	140	Olivier	01/05/22	A
2	90	Coliseum	04/06/22	B
2	220	Lyttelton	08/03/22	VIP
3	140	Olivier	14/04/22	B
3	220	Olivier	20/03/22	VIP

There are no candidate key with one attribute

Candidate keys

BUYER_ID	PRICE	HALL_NO	DATE	TYPE
1	150	Coliseum	08/03/22	VIP
1	150	Lyttelton	14/04/22	A
2	140	Olivier	01/05/22	A
2	90	Coliseum	04/06/22	B
2	220	Lyttelton	08/03/22	VIP
3	140	Olivier	14/04/22	B
3	220	Olivier	20/03/22	VIP

Unique keys: ~~(BUYER_ID, PRICE)~~

Candidate keys

BUYER_ID	PRICE	HALL_NO	DATE	TYPE
1	150	Coliseum	08/03/22	VIP
1	150	Lyttelton	14/04/22	A
2	140	Olivier	01/05/22	A
2	90	Coliseum	04/06/22	B
2	220	Lyttelton	08/03/22	VIP
3	140	Olivier	14/04/22	B
3	220	Olivier	20/03/22	VIP

Unique keys: ~~(BUYER_ID, HALL_NO)~~

Candidate keys

BUYER_ID	PRICE	HALL_NO	DATE	TYPE
1	150	Coliseum	08/03/22	VIP
1	150	Lyttelton	14/04/22	A
2	140	Olivier	01/05/22	A
2	90	Coliseum	04/06/22	B
2	220	Lyttelton	08/03/22	VIP
3	140	Olivier	14/04/22	B
3	220	Olivier	20/03/22	VIP

Unique keys: (BUYER_ID, DATE) (BUYER_ID, VIP) (PRICE, DATE) (HALL_NO, DATE) (HALL_NO, TYPE)
(BUYER_ID, PRICE, HALL_NO) (BUYER_ID, HALL_NO, DATE)
etc -- all sets with 3, 4 or 5 attributes

Candidate keys

BUYER_ID	PRICE	HALL_NO	DATE	TYPE
1	150	Coliseum	08/03/22	VIP
1	150	Lyttelton	14/04/22	A
2	140	Olivier	01/05/22	A
2	90	Coliseum	04/06/22	B
2	220	Lyttelton	08/03/22	VIP
3	140	Olivier	14/04/22	B
3	220	Olivier	20/03/22	VIP

Candidate keys: (BUYER_ID, DATE) (BUYER_ID, VIP) (PRICE, DATE) (HALL_NO, DATE) (HALL_NO, TYPE)
(BUYER_ID, PRICE, HALL_NO) (BUYER_ID, PRICE, TYPE) ,
~~(PRICE, DATE, DATE, TYPE)~~

Candidate keys

BUYER_ID	PRICE	HALL_NO	DATE	TYPE
1	150	Coliseum	08/03/22	VIP
1	150	Lyttelton	14/04/22	A
2	140	Olivier	01/05/22	A
2	90	Coliseum	04/06/22	B
2	220	Lyttelton	08/03/22	VIP
3	140	Olivier	14/04/22	B
3	220	Olivier	20/03/22	VIP

Candidate keys: (BUYER_ID, DATE) (BUYER_ID, VIP) (PRICE, DATE) (HALL_NO, DATE) (HALL_NO, TYPE)
(BUYER_ID, PRICE, HALL_NO) (BUYER_ID, PRICE, TYPE) ,

Participant
participant_id last_name first_name

Product
product_id name code reserve_price

Info
photo description

Bid
price timestamp participant_id product_id currency

candidate
keys?

Category
category_id name

<<enum>> Currency
USD EUR GBP ...

Please answer www.menti.com 44 32 42 0

Q8

Primary key

- **Unique** identifier.
 - **not null** must be known at any moment.
 - Simple, no ambiguities.
 - Stable.
 - Must be a candidate key
-
- Composed keys may be replaced with an *artificial key (surrogate key)*.
 - In many RDBMS we may use autoincremented values.

Primary key UUID/GUID

- **universally unique identifier** 128-bit
 - Probability of collision (that a UUID is duplicated) is negligible.
 - No need to change when merging to databases. (example: store on local database, merge when connected to central database)
 - Known before the insertion of a new row, without querying the database.
-
- Please answer [www.menti.com](https://www.menti.com/join/80457859) 8045 7859 Q7

Primary key

- Undersize/oversize?.
 - Has foreign keys? (how many lines are affected?)
 - Is it stable?
 - Is it simple?
-
- Please answer [www.menti.com](https://www.menti.com/join/80457859) 8045 7859 Q7

Populating a Primary Key

- Identity automatically assigns a unique sequence number to each row inserted.
- Sequence or functions (functions that generate uuid).
- Examples:
 - Postgres SERIAL (populate with sequence values)
 - MySql AUTO_INCREMENT.
 - Oracle Sequence
 - My Sql Server IDENTITY
 - Mongo ObjectID -- UUID

Primary key UUID/GUID

- **universally unique identifier** 128-bit
- Not the best solution for clusters (sequential UUIDs might be used).
- Types:
 - Type 1 : 4 bytes + 2 bytes + 2 bytes + 2 bytes + 6 bytes = time + node
 - Type 4 : 122 bits randomly generated, 6 bits reserved for version and variant.
- Bit for type
 - type 1 2ad1db02-2ff0-11eb-**a**dc1-0242ac120002
 - type 4 a7bc2d72-7153-44a1-**8**3df-d03dd298cf53

Primary key UUID/GUID

- **universally unique identifier** 128-bit
- Not the best solution for clusters (sequential UUIDs might be used).
- Types:
 - Type 1 : 4 bytes + 2 bytes + 2 bytes + 2 bytes + 6 bytes = time + node
 - Type 4 : 122 bits randomly generated, 6 bits reserved for version and variant.
- Bit for type
 - type 1 2ad1db02-2ff0-11eb-**a**dc1-0242ac120002
 - type 4 a7bc2d72-7153-44a1-**8**3df-d03dd298cf53

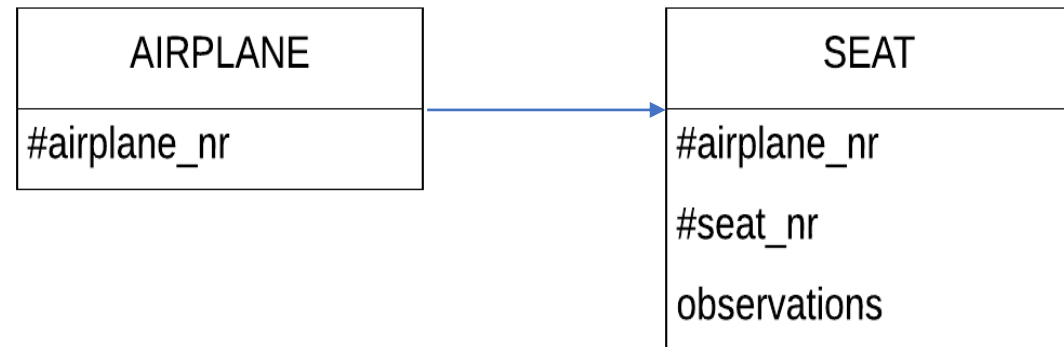
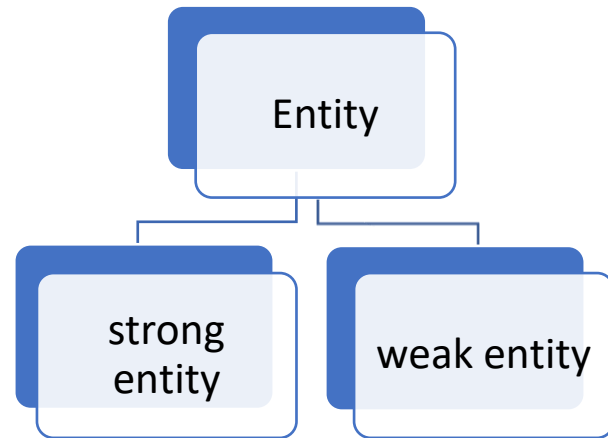
Airline (1)Entities

- The airline has one or more airplanes. An airplane has a model number, and capacity. Each flight is carried out by airplanes. An airplane is uniquely identified by its Registration_no and a flight is identified by its Flight_no. A passenger can book a ticket for a flight.

Please answer www.menti.com 8045 7859

Q10, Q11

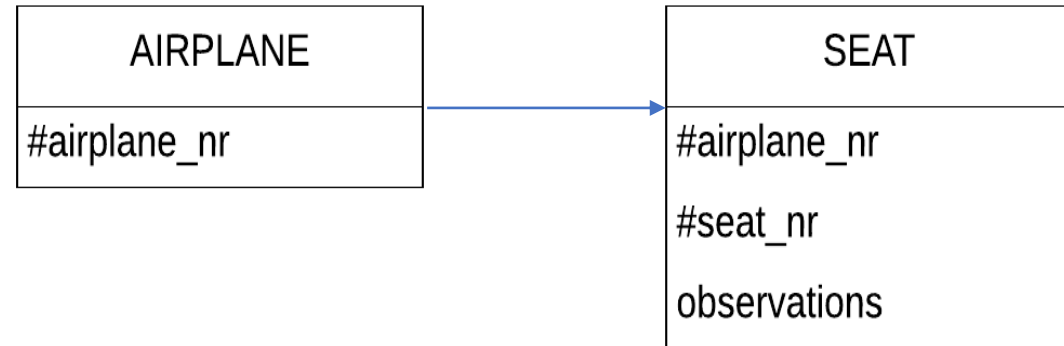
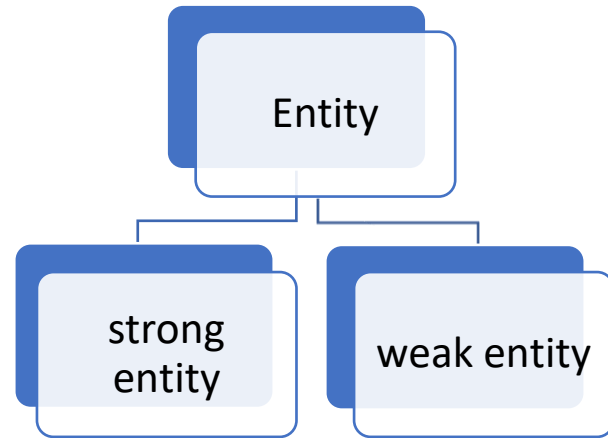
Entities



- Weak entity is an entity that depends on another entity.
- The primary key of a weak entity contains the primary key of the strong entity that it depends on + description/partial key.

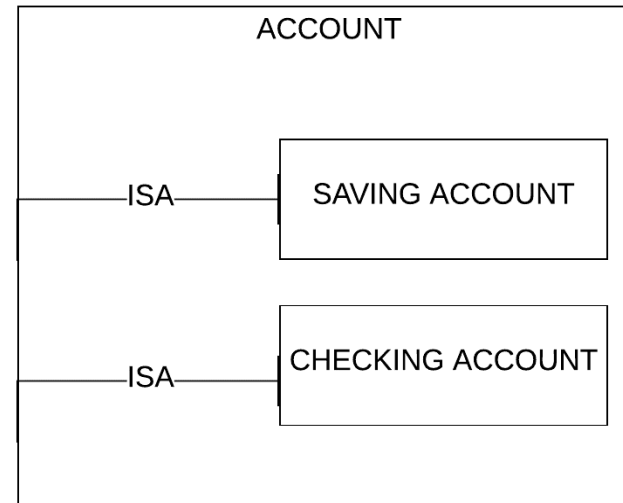
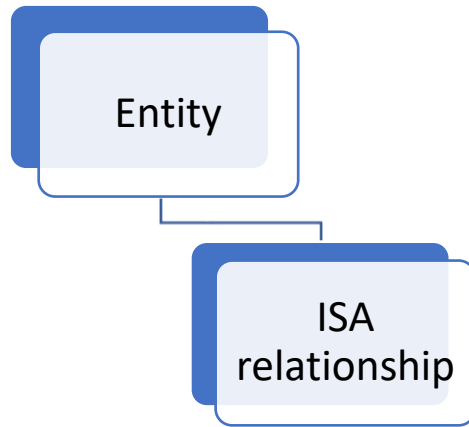
Please answer [www.menti.com](https://www.menti.com/join/80457859) 8045 7859 Q12, 13

Entities



- Weak entity is an entity that depends on another entity.
- The primary key of a weak entity contains the primary key of the strong entity that it depends on + description/partial key.

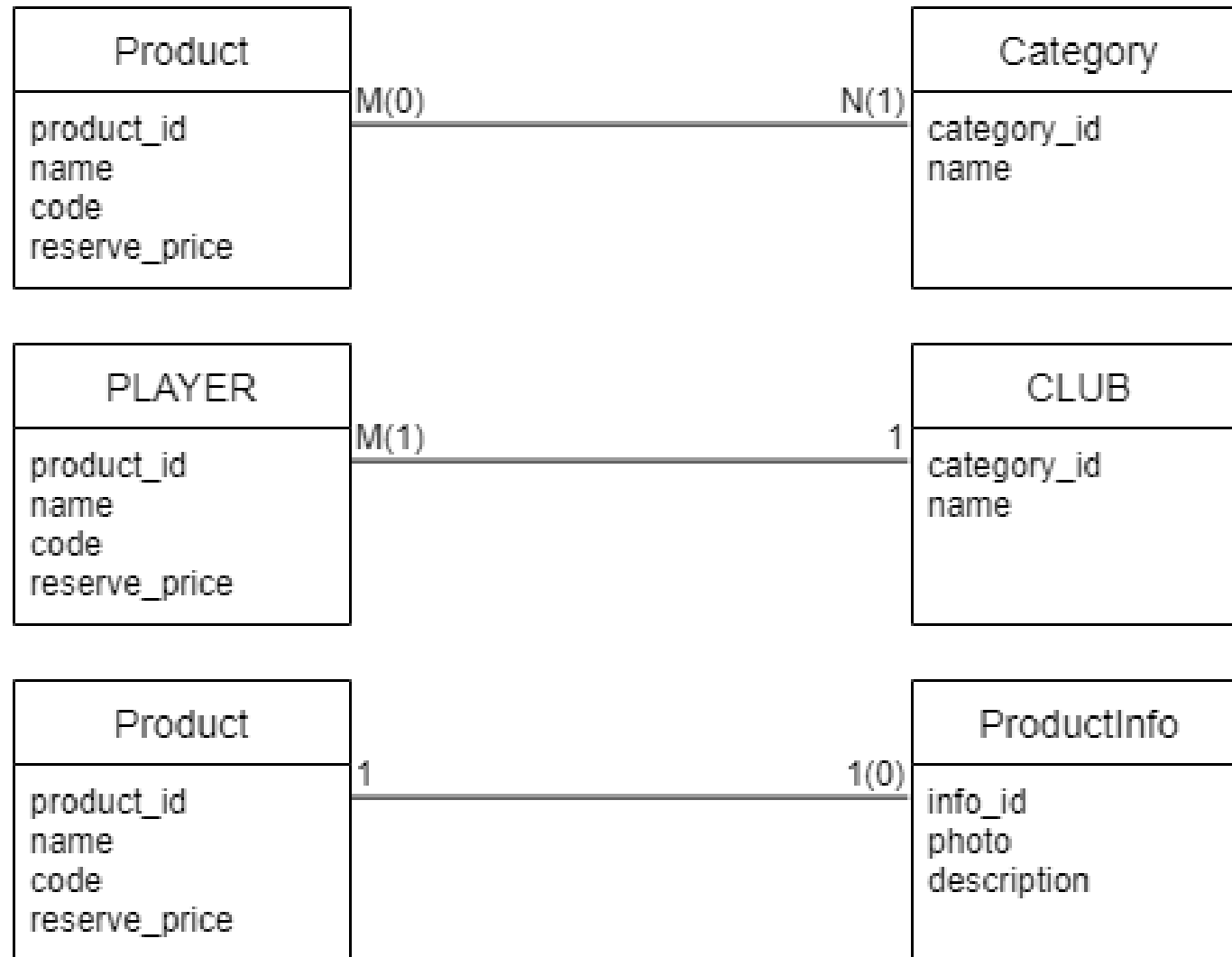
Entities



- A sub-entity has the same key as the *super*-entity and all its attributes and relationships.

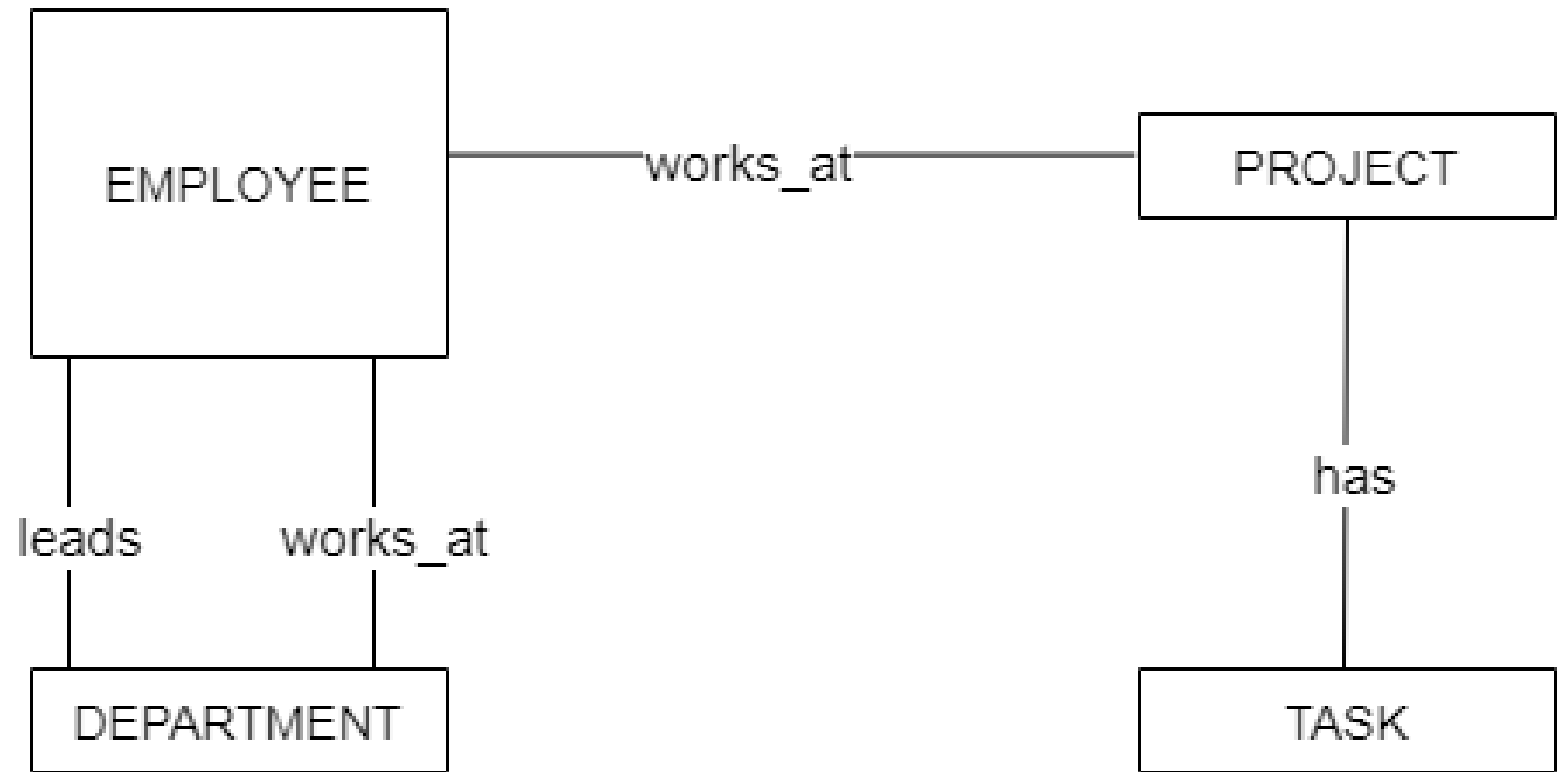
Relationship

- Association between two or more entities (binary, ternary etc.)
- Relationship → column (foreign key) or table.
- Graphical representation: oriented arc.
- Two relationships with the same name link different entities.
- Cardinality defines the numerical attributes of the relationship between two entities: **MANDATORY** (min) **OPTIONAL** (max)



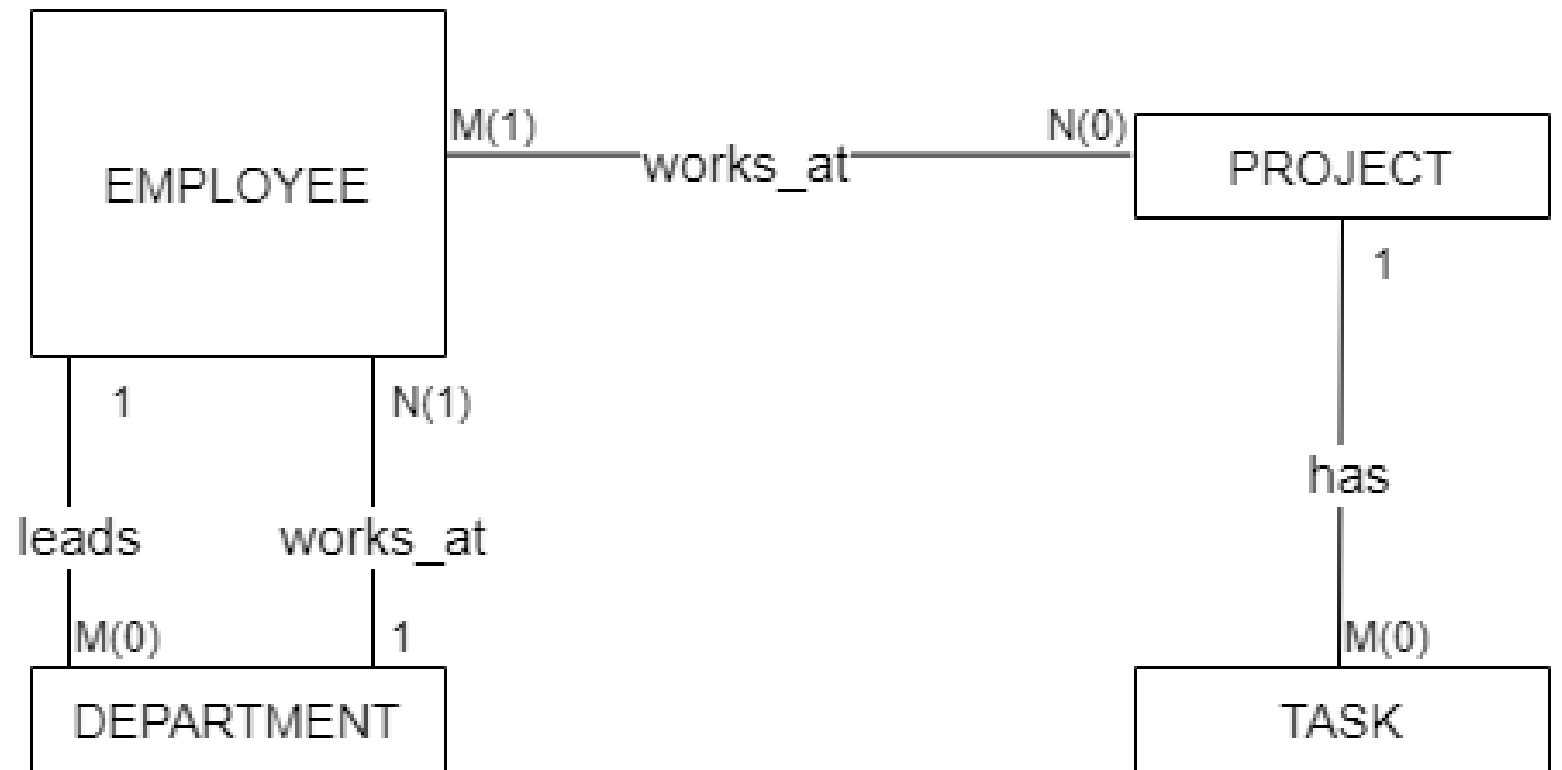
Relationship cardinality

- MANDATORY (must)
- OPTIONAL (may)



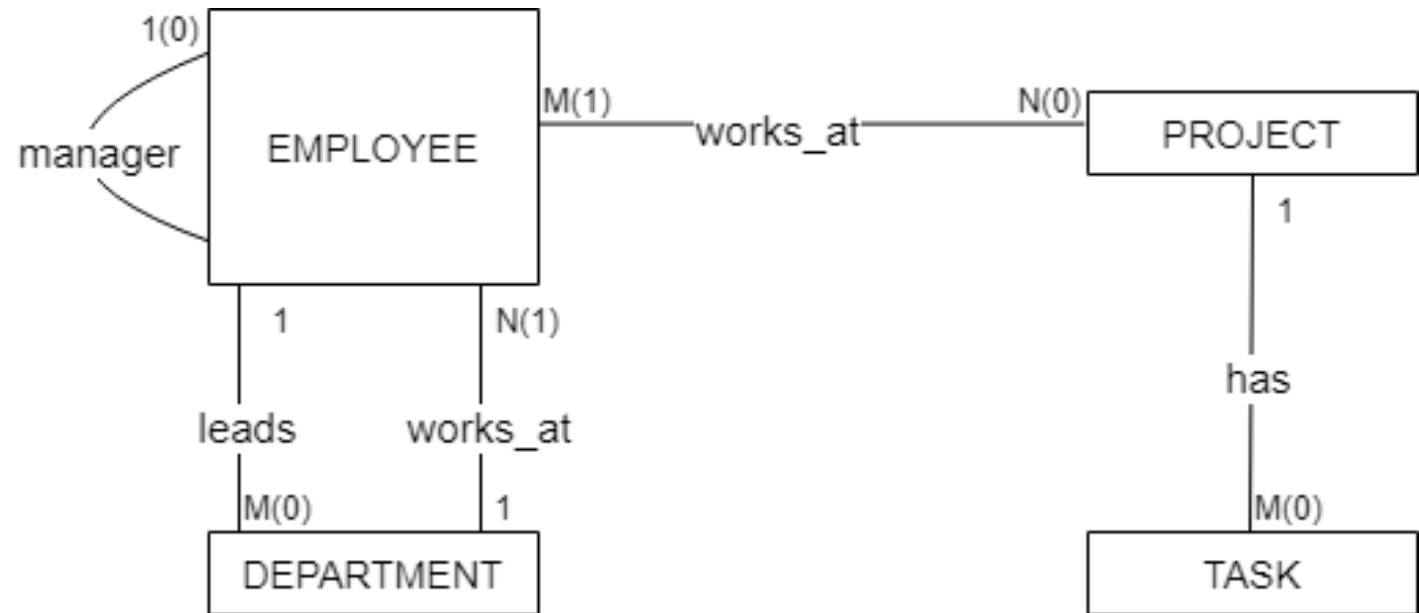
Relationship cardinality

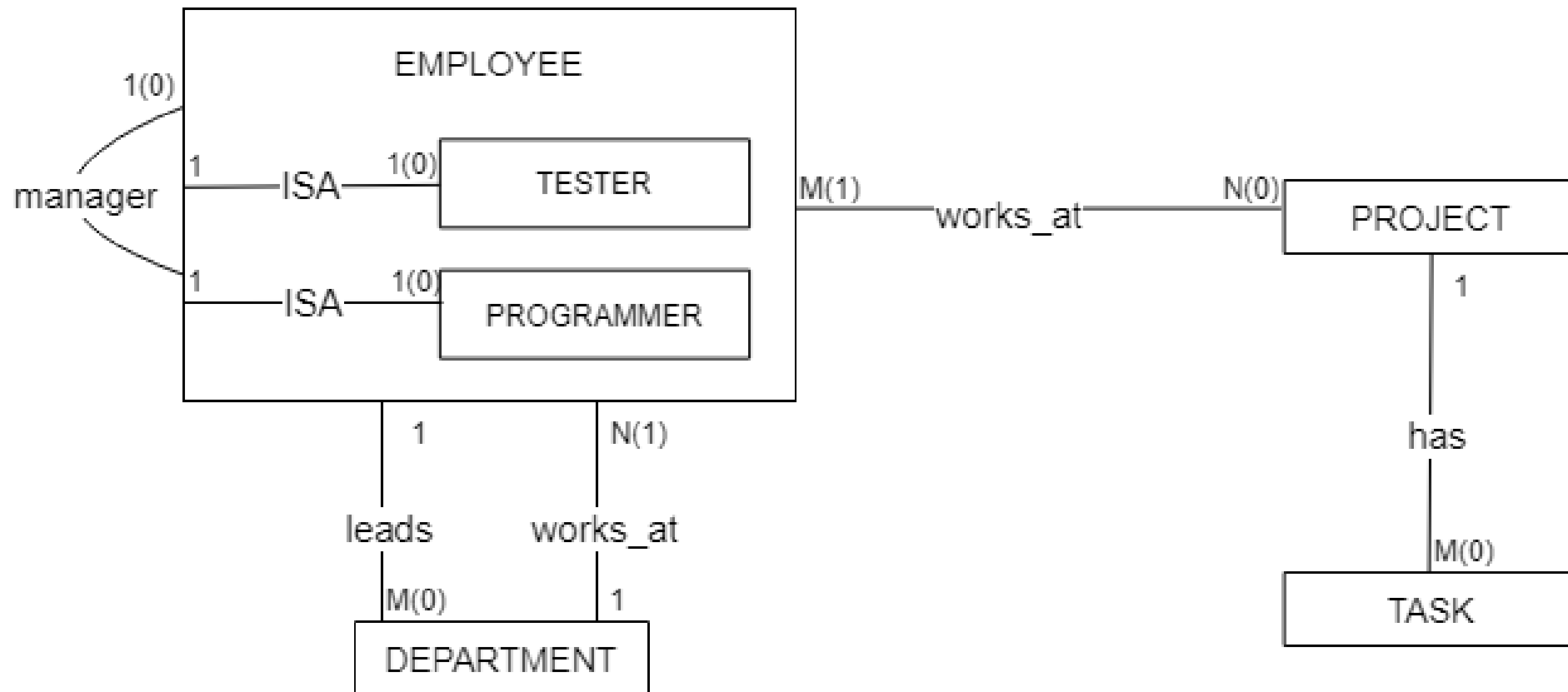
- MANDATORY (must)
- OPTIONAL (may)



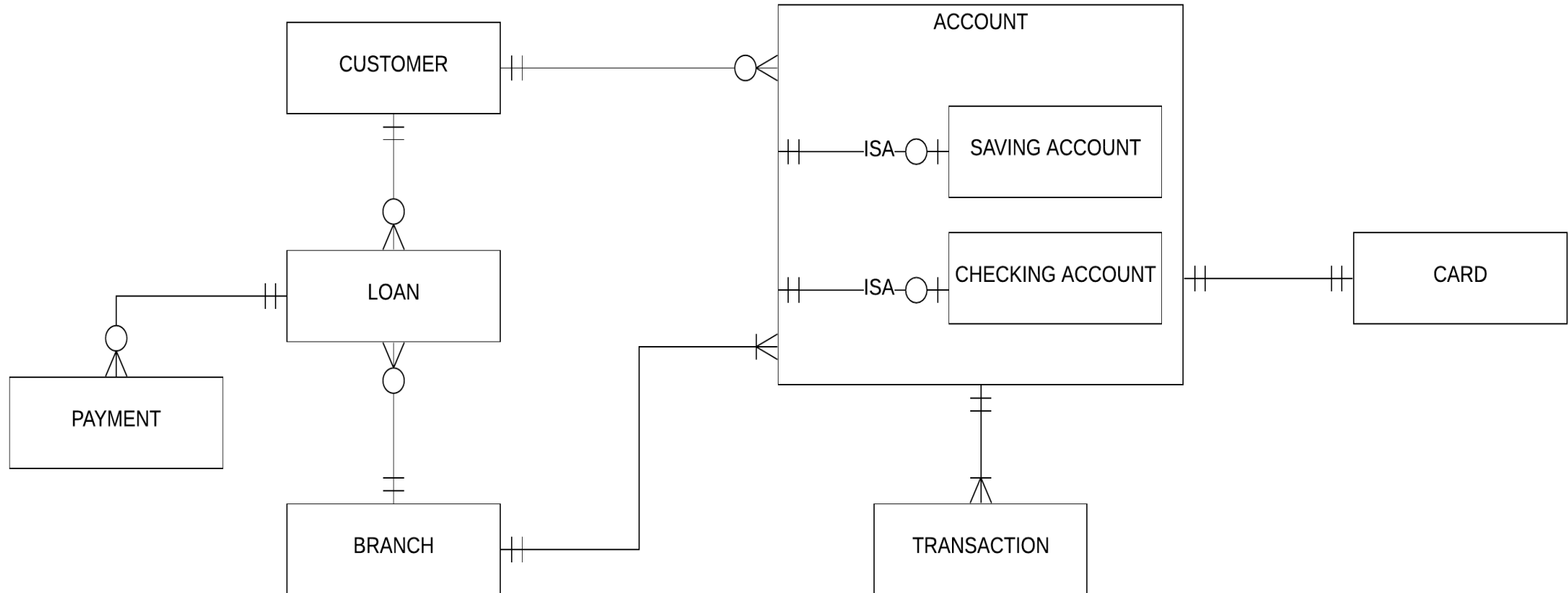
Relationship cardinality

- Reflexive relationship
unary relationship.





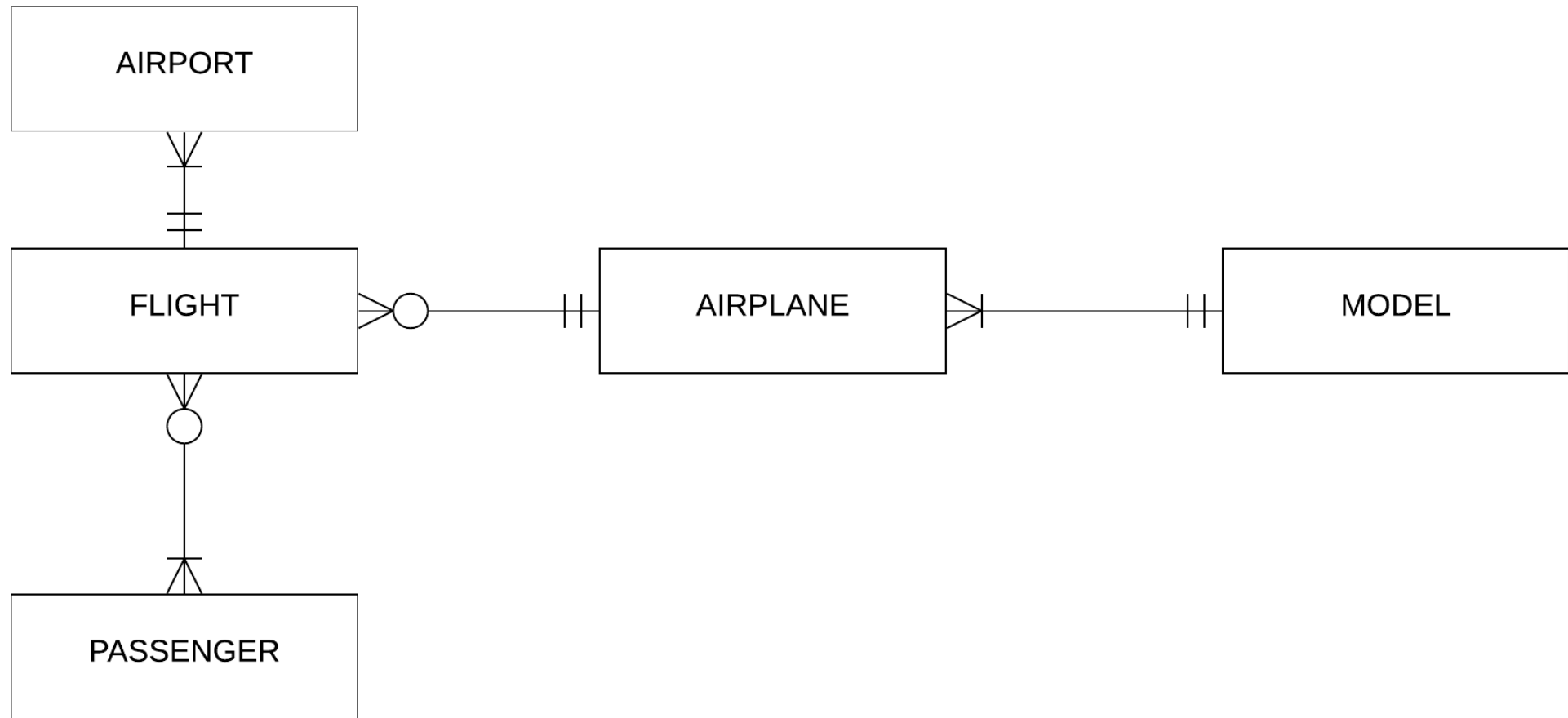
Banking Relationships



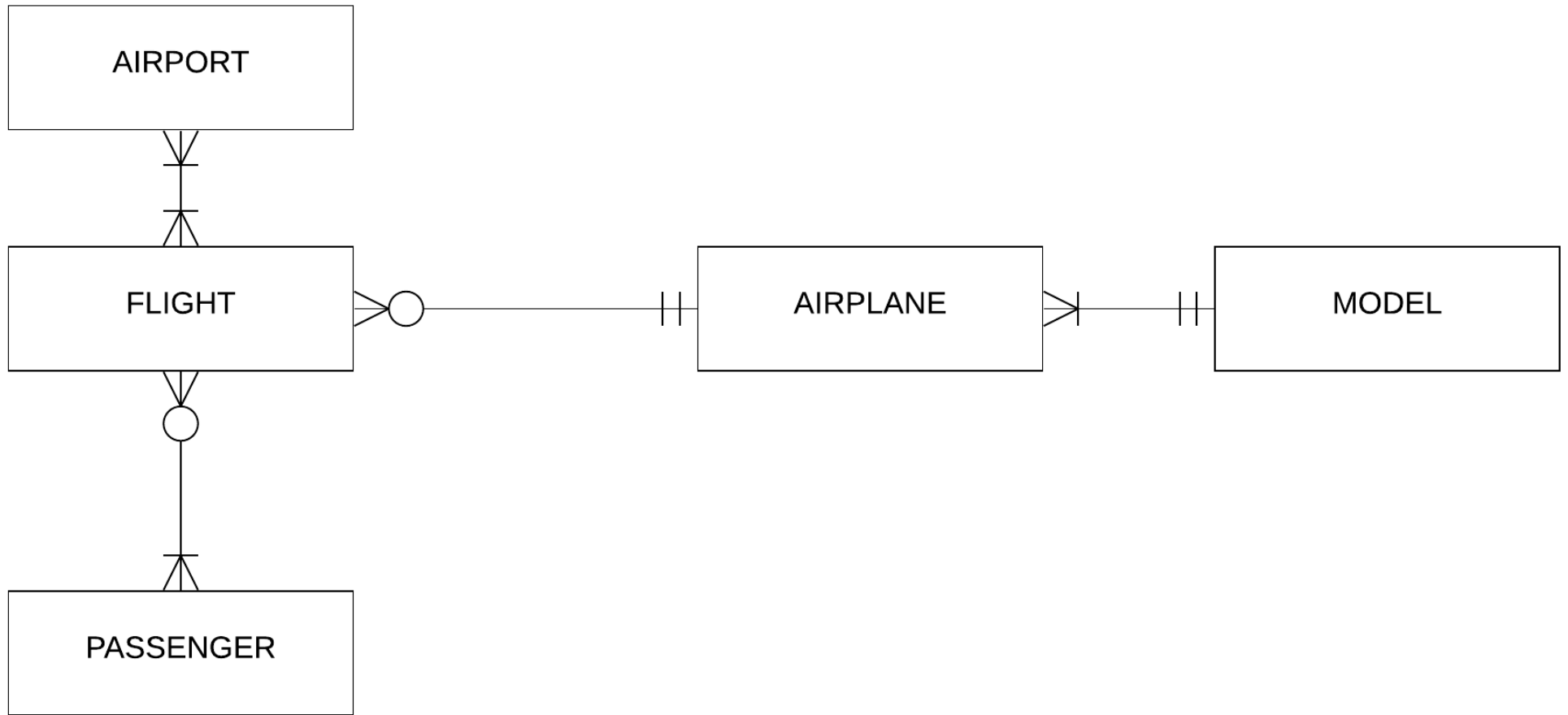
Airline Relationships

- The airline has one or more airplanes. An airplane has a model number, and capacity. Each flight is carried out by airplanes. An airplane is uniquely identified by its Registration_No and a flight is identified by its Flight_No. A passenger can book a ticket for a flight.

Airline



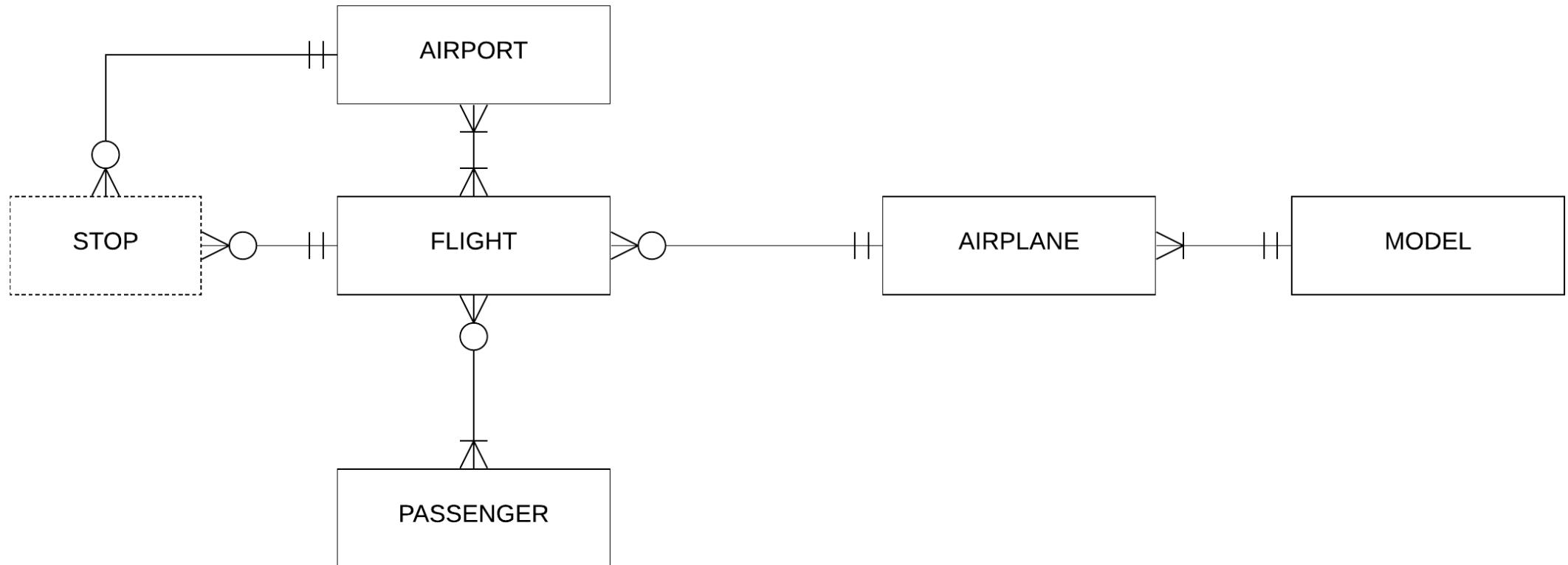
Airline



Airline Relationships

- The airline has one or more airplanes. An airplane has a model number, and capacity. Each flight is carried out by airplanes. An airplane is uniquely identified by its Registration_No and a flight is identified by its Flight_No. A passenger can book a ticket for a flight. A flight may have **one or more stops**.

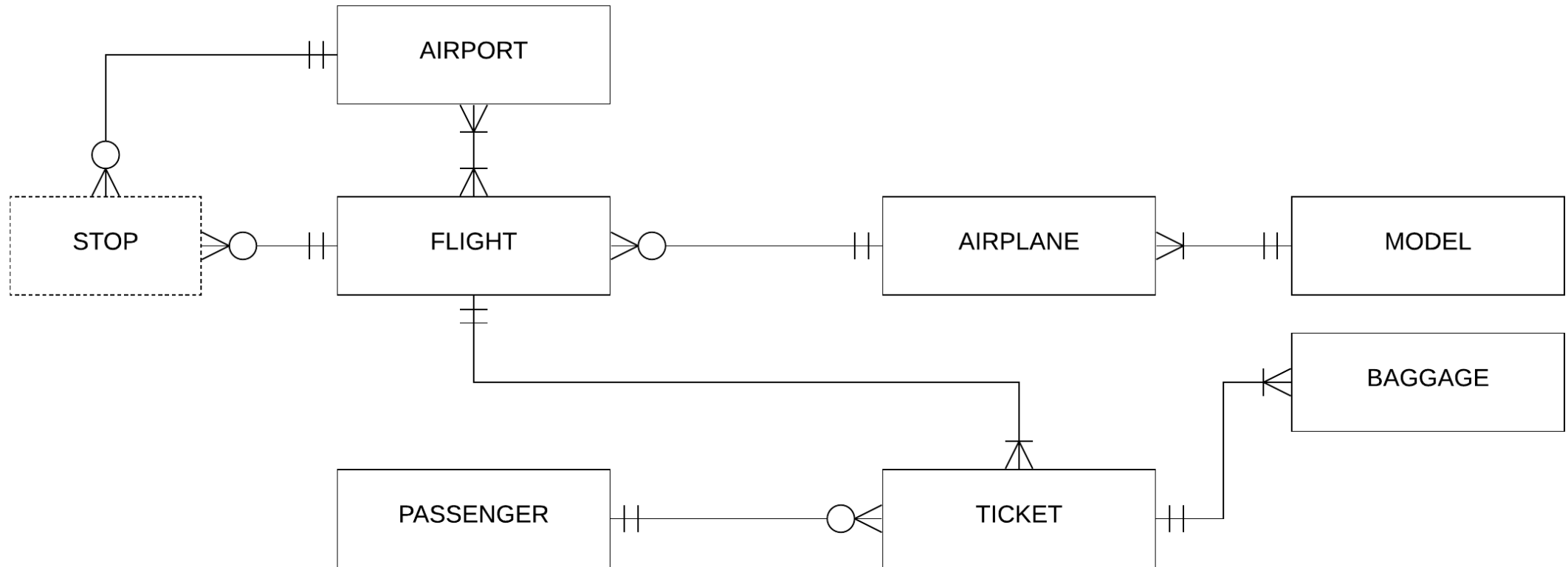
Airline



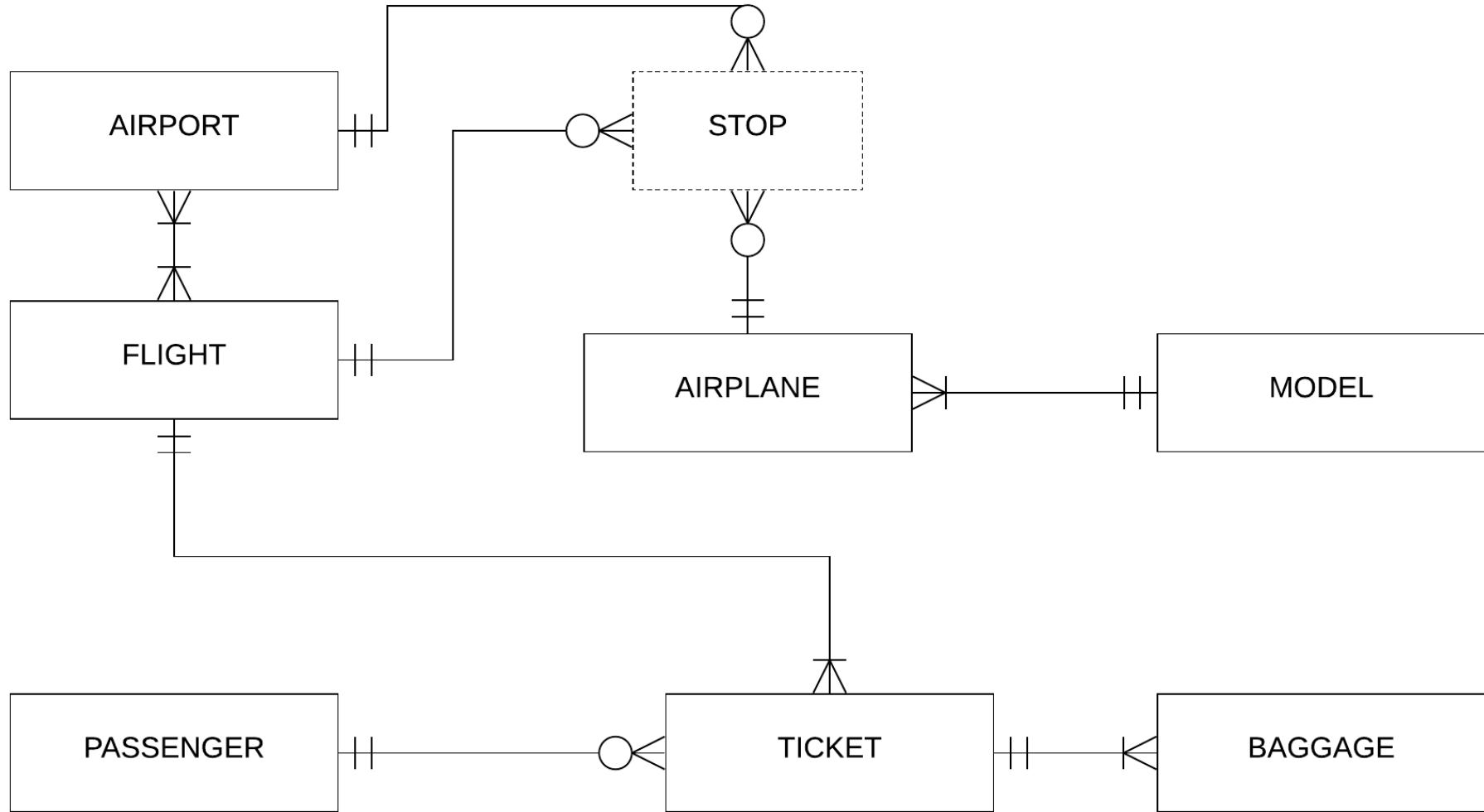
Airline Relationships

- The airline has one or more airplanes. An airplane has a model number, and capacity. Each flight is carried out by airplanes. An airplane is uniquely identified by its Registration_No and a flight is identified by its Flight_No. A passenger can book a ticket for a flight. A flight may have one or more stops. The passenger will pay for **extra baggage**.

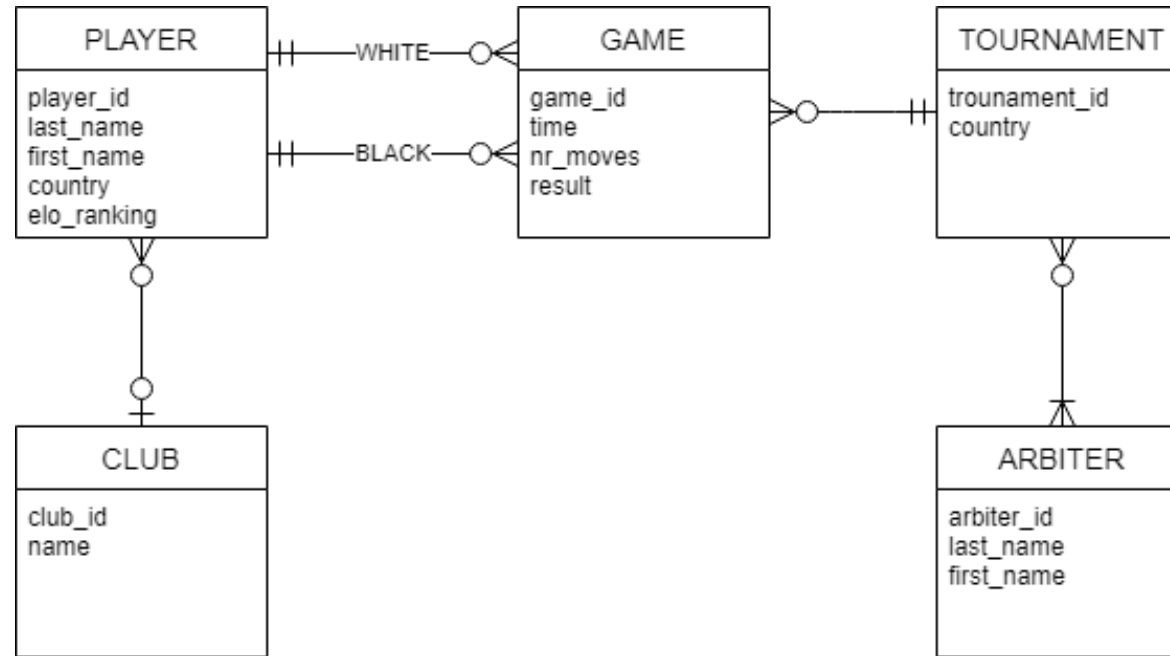
Airline



Airline

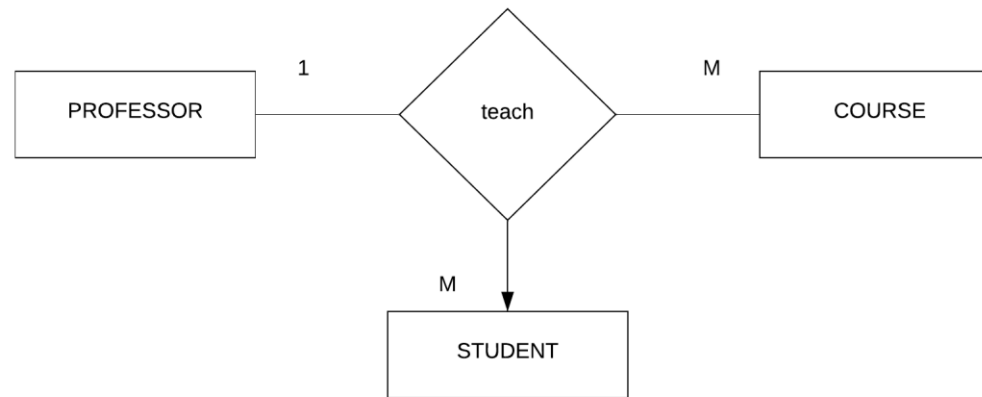


Chess



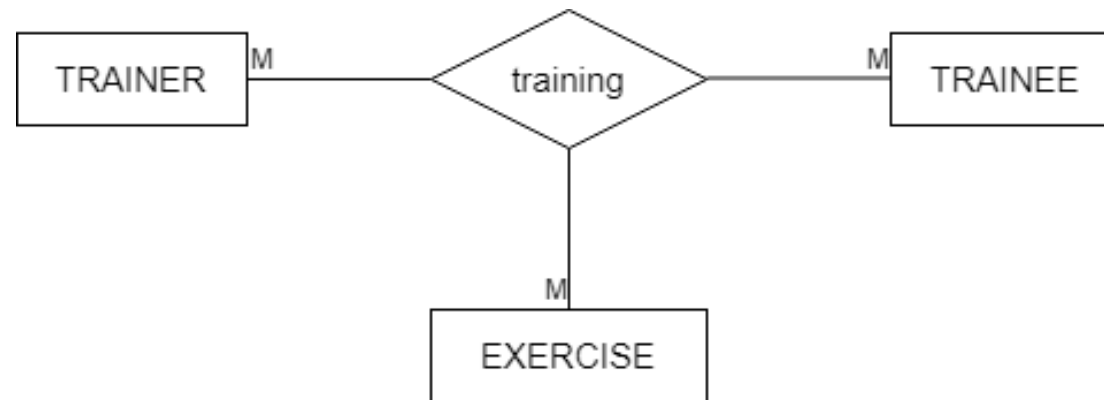
Ternary relationships

- Relationship binding simultaneously 3 entities.



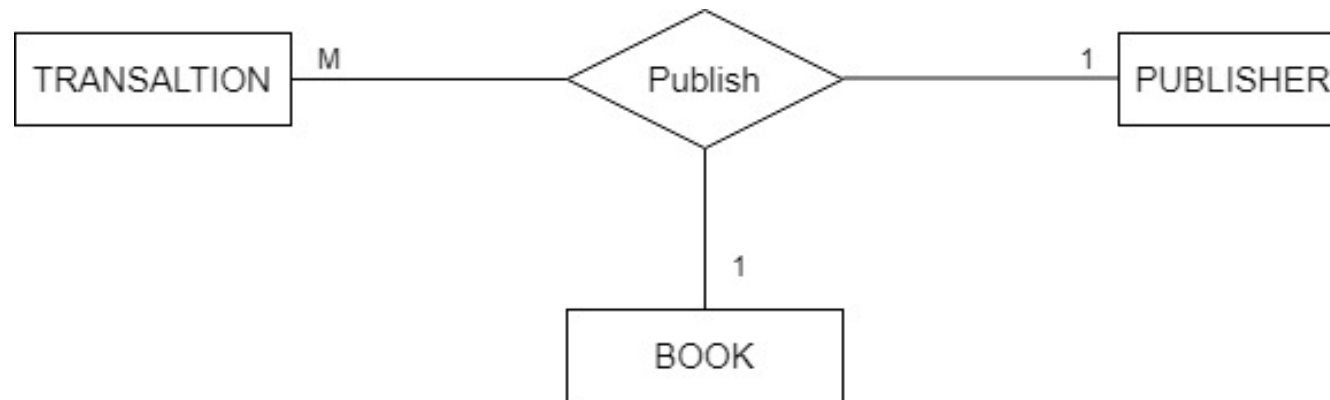
Ternary relationships

- Relationship binding simultaneously 3 entities.



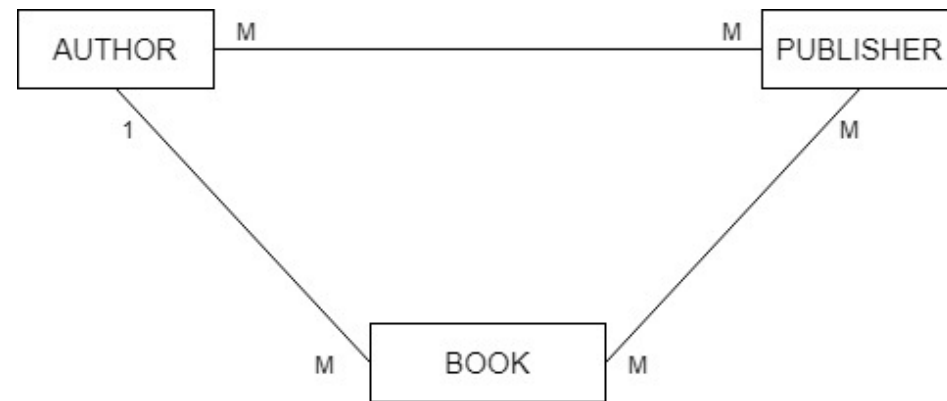
Ternary relationships

- Ternary or three binary?



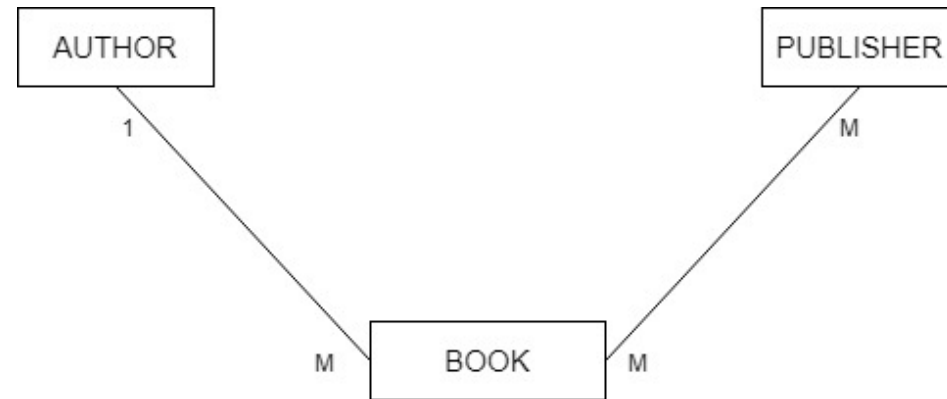
Ternary relationships

- Ternary or three binary?



Ternary relationships

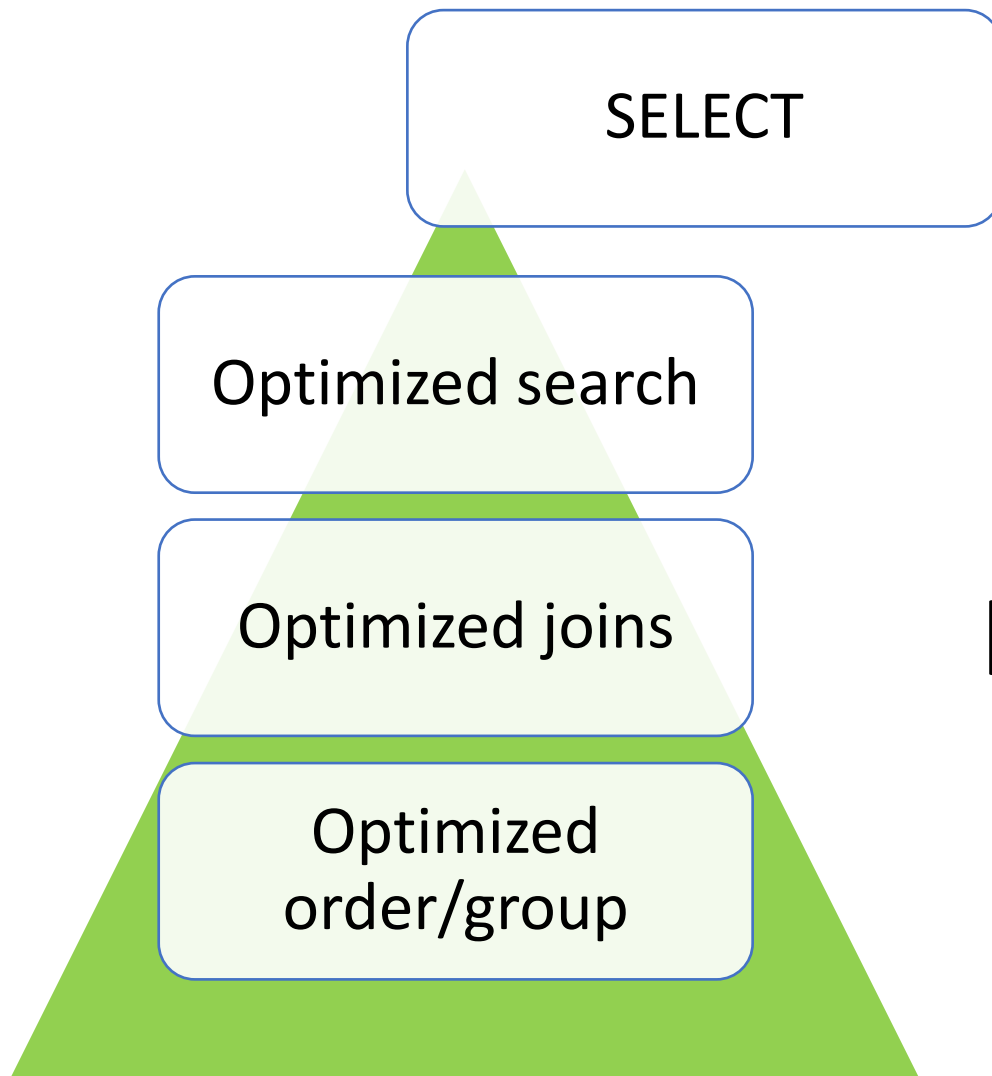
- Ternary or three binary?



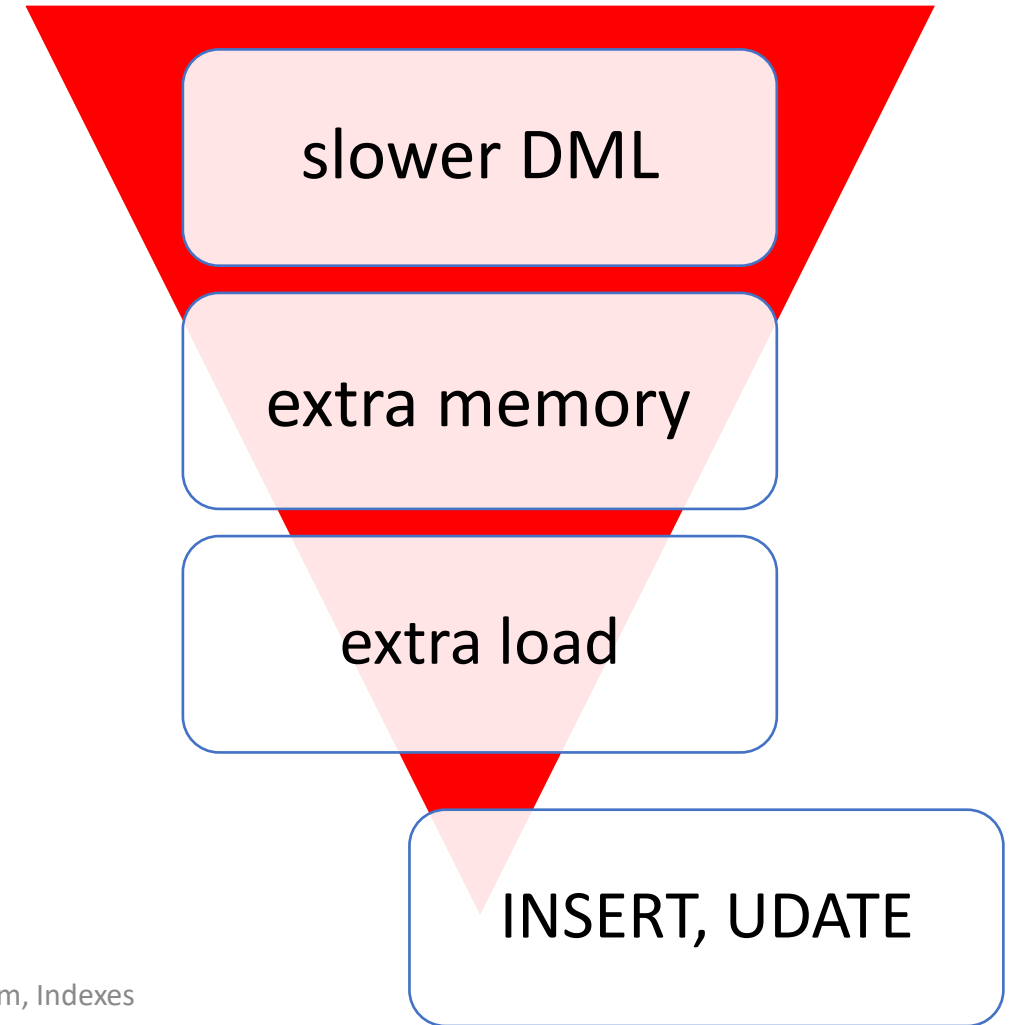
Indexes

Indexes

- Maps search key to data using specific data structures.
- Optimized search.
- Optimized joins (lookup in more than one table)
- Optimized order/group
- slower DML (insert and update operations).
- extra memory



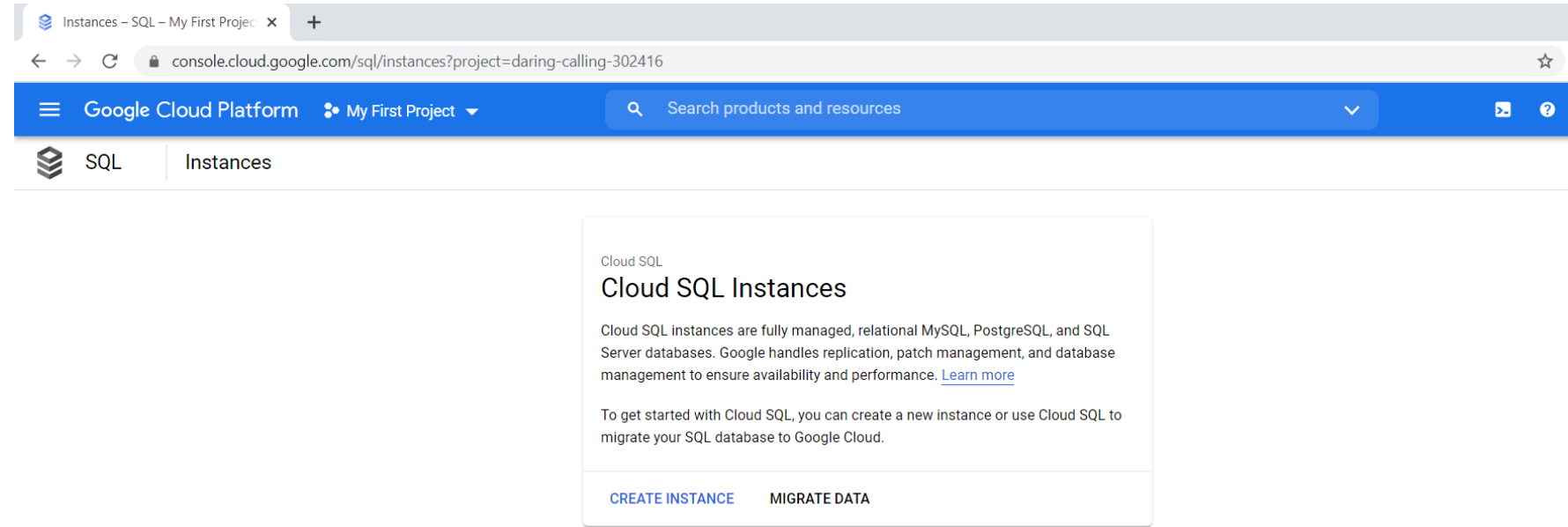
Index



Indexes demo

In Google Cloud go to
**SQL – CREATE
INSTANCE.**

Choose MySQL.
PostgreSQL and
SQLServer are also
available.

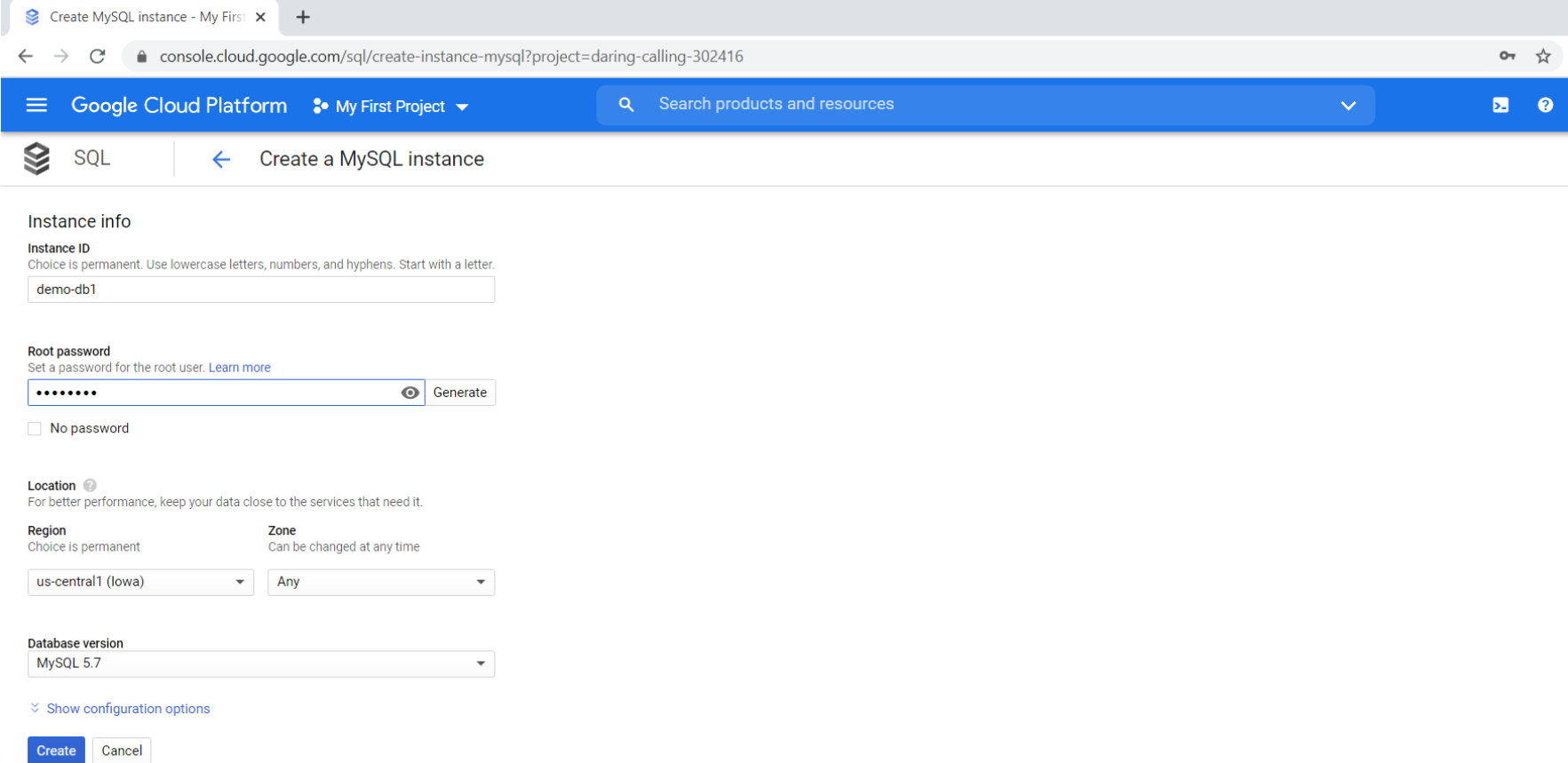


Indexes demo

Choose Instance ID and root password.

Wait for the instance to be created.

Connects with Cloud Shell and Cloud Proxy are available.



Create MySQL instance - My First x +

console.cloud.google.com/sql/create-instance-mysql?project=daring-calling-302416

Google Cloud Platform My First Project Search products and resources

SQL Create a MySQL instance

Instance info

Instance ID
Choice is permanent. Use lowercase letters, numbers, and hyphens. Start with a letter.

demo-db1

Root password
Set a password for the root user. [Learn more](#)

..... Generate

☐ No password

Location ?
For better performance, keep your data close to the services that need it.

Region
Choice is permanent

us-central1 (Iowa)

Zone
Can be changed at any time

Any

Database version

MySQL 5.7

Show configuration options

Create Cancel

Indexes demo

Enable Cloud Admin and try connecting with cloud console.

To connect with Google Proxy follow these steps:

<https://cloud.google.com/sql/docs/mysql/connect-admin-proxy>

The screenshot displays the Google Cloud Platform console for a Cloud SQL instance named 'demo-db1'. The left sidebar shows the 'Overview' tab selected under the 'SQL' section. The main content area includes a 'Connect to this instance' section with the public IP address '34.122.6.43' and a 'Configuration' section showing 1 vCPU, 3.75 GB Memory, and 10 GB SSD storage. Below the main content, a 'CLOUD SHELL' terminal window is open, showing a successful connection to the MySQL database using the 'gcloud sql connect' command. The terminal output includes the MySQL welcome message and the result of a 'select sysdate()' query, which returns '2021-01-21 17:01:33'.

```
iulia_th84@cloudshell:~ (daring-calling-302416)$ gcloud sql connect demo-db1 --user=root --quiet
Allowlisting your IP for incoming connection for 5 minutes...done.
Connecting to database with SQL user [root].Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 66
Server version: 5.7.25-google-log (Google)

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> select sysdate()
-> ;
+-----+
| sysdate() |
+-----+
| 2021-01-21 17:01:33 |
+-----+
1 row in set (0.11 sec)

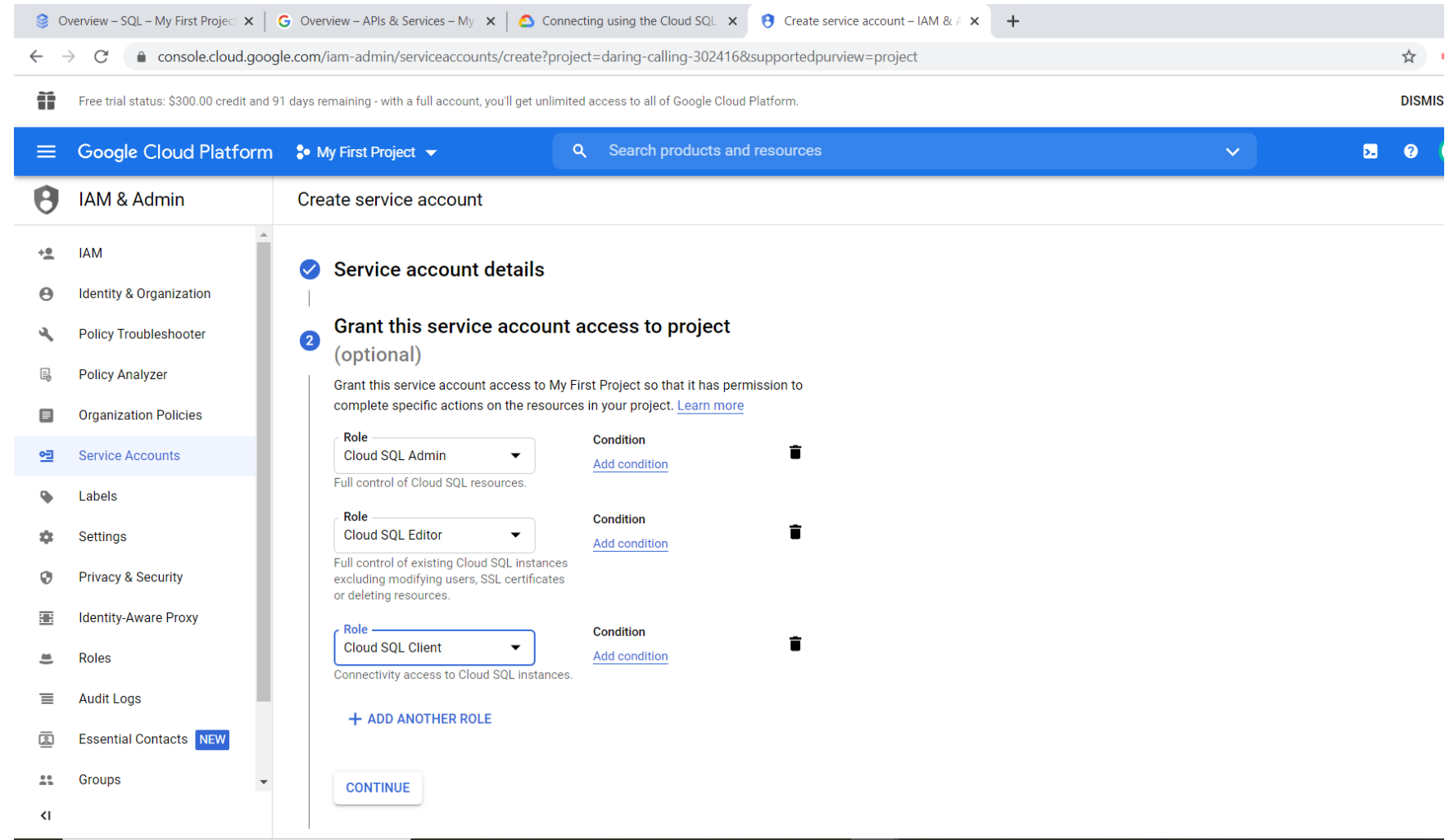
mysql>
```

Indexes demo

To connect with Google Proxy follow these steps:

<https://cloud.google.com/sql/docs/mysql/connect-admin-proxy>

Create a service account and store the key.



Indexes demo

Start cloud_sql_proxy:

```
./cloud_sql_proxy -
instances=daring-
calling-302416:us-
central1:demo-
db1=tcp:3306 -
credential_file="./daring-
calling-302416-
940c046116a7.json"
```

Connect with MySQL Workbench with root user and run **mysqlCreateDB.sql**.

The screenshot displays a Windows PowerShell terminal window at the top, showing the execution of the `cloud_sql_proxy` command to establish a connection to a Google Cloud SQL instance. The command is: `PS C:\LABORATOARE\BD2020\googleCloud> ./cloud_sql_proxy -instances=daring-calling-302416:us-central1:demo-db1=tcp:3306 -credential_file="./daring-calling-302416-940c046116a7.json"`. The terminal output indicates that the proxy is listening on port 3306 and has successfully established a new connection for the specified instance.

Below the terminal is the MySQL Workbench interface. The left sidebar shows the 'MANAGEMENT' section with options like 'Server Status', 'Client Connections', and 'Users and Privileges'. The 'INSTANCE' section includes 'Startup / Shutdown', 'Server Logs', and 'Options File'. The 'PERFORMANCE' section includes 'Dashboard' and 'Performance Reports'. The 'Administration' tab is selected, showing 'Schemas' and 'Information'.

The main editor area shows a SQL script with the following content:

```
1  /*https://cloud.google.com/solutions/setup-mysql
2
3  https://cloud.google.com/sql/docs/mysql/quickstart
4
5  create database demo*/
6
7
8  • create database demo;
9  • use demo;
10
11 /*select user from mysql.user;*/
12
13 • create user 'user_demo' identified by 'user_demo';
14
15 • grant all privileges on demo.* to user_demo with grant option;
16
```

Indexes demo

Connect with MySQL Workbench with user **demo** and run **mySqlCreqteDemoTables.sql**

and **C2AddIndexCompareTimes.sql**

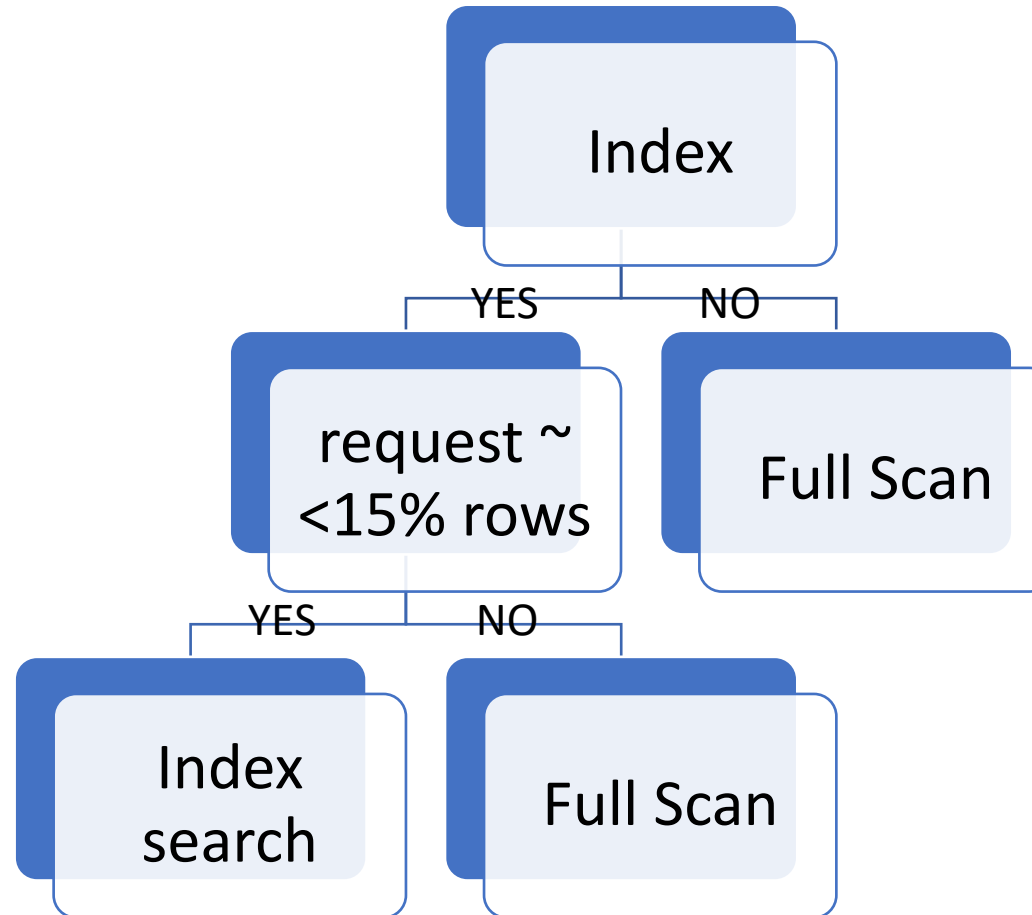
The screenshot shows the MySQL Workbench interface with the following components:

- Navigator:** Shows the 'demo' schema with tables 'no_index' and 'with_index'.
- SQL Editor:** Contains the following SQL code:

```
1 CREATE TABLE no_index (  
2     id int,  
3     uuid VARCHAR(36) NOT NULL,  
4     random_number VARCHAR(10)  
5 );  
6  
7 CREATE TABLE with_index (  
8     id int,  
9     uuid VARCHAR(36) NOT NULL,  
10    random_number VARCHAR(10)  
11 );  
12  
13 delimiter $$  
14 CREATE procedure `insert_test_function` ()  
15 BEGIN  
16     DECLARE i integer;  
17     DECLARE suuid varchar(36);  
18     DECLARE randnum varchar(10);  
19  
20     set i = 1;  
21     WHILE i <= 25000000 DO  
22         set suuid = uuid();
```
- Output:** Shows the execution results of the queries.

#	Time	Action	Message	
✓ 18	14:06:31	CREATE TABLE no_index (id int, uuid VARCHAR(36) NOT NULL, random_number VARCHAR(...	0 row(s) affected	
✓ 19	14:06:31	CREATE TABLE with_index (id int, uuid VARCHAR(36) NOT NULL, random_number VARCHA...	0 row(s) affected	
✓ 20	14:06:32	CREATE procedure `insert_test_function` () BEGIN DECLARE i integer; DECLARE suuid varchar(36); ...	0 row(s) affected	
4	21	14:06:32	call `demo`.`insert_test_function`()	Running...

Sql Optimizer



Autogenerated columns

- MySQL auto-generated index (key):
 - DB_ROW_ID increases monotonically as new rows are inserted.
 - DB_ROLL_PTR roll pointer, points to log record.
 - DB_TRX_ID last transaction that updated or inserted the row.
- Oracle rowid:
 - Pseudo column 18 characters = 10 + 4 + 4 (block, row, file).
 - Store and return row address in hexadecimal format (string).
 - Unique identifier for each row.
 - Immutable.

Autogenerated columns

- Oracle rowid:
 - Used in where clause to select/update/delete a row.
- Oracle rownum:
 - Sequential number in which oracle has fetched the row, before ordering the result
 - Temporary generated along with a select statement.
- Mongo
 - ObjectID (timestamp 4Bytes + random 5Bytes + Count 3Bytes).

Index

- Data structure that optimize search.
- Automatically created when a primary key is defined.

MySQL

```
SHOW EXTENDED INDEX FROM index_test;
```

Oracle

```
select * from user_indexes  
where table_name = 'INDEX_TEST';
```

Primary key

- Constraint imposed on insert/update behavior.
- NotNull & Unique.

MySQL

```
select * from information_schema.statistics  
where table_name = 'index_test1'  
and index_name = 'primary';
```

Oracle

```
select * from user_constraints  
where table_name = 'INDEX_TEST';
```