ER - DIAGRAM

COURSE 1: Databases

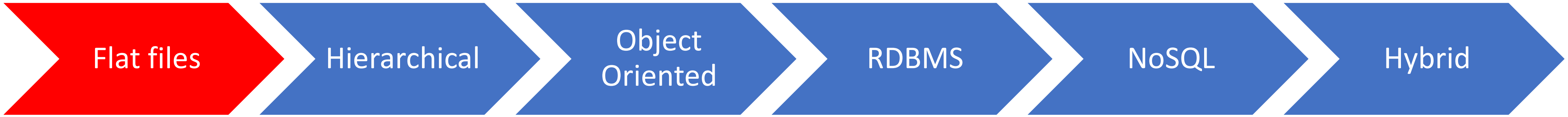
DBMS (Database Management System)

- System designed to define and manipulate data.
    - Storage.
    - Retrieval.
    - Updates.

DBMS (Database Management System)

- Avoid redundancy, inconsistency.
- Concurrent data access.
- Provides security and recovery.
- Declarative language to manipulate, query, define and control data.
- DDL, DML, DCL.
- Data dictionary: database providing info about database structure.

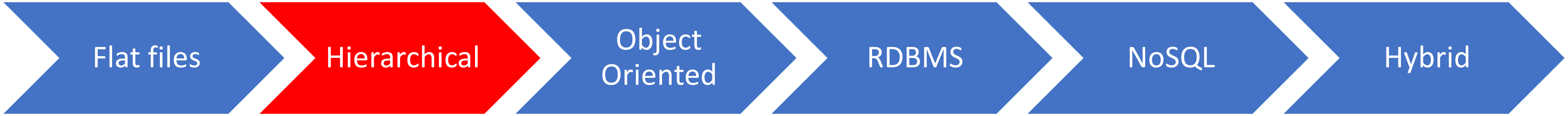
DBMS (Database Management System)



Flat files → Hierarchical → Object Oriented → RDBMS → NoSQL → Hybrid

- Text database, example CSV format.
- Implemented in 1970 (IBM).
- File = table with a single record on each line.
- Read, store and send.
- Simple structure.
- Inefficient: slow, duplicated values, hard to update etc.

DBMS (Database Management System)



Flat files

Hierarchical

Object Oriented

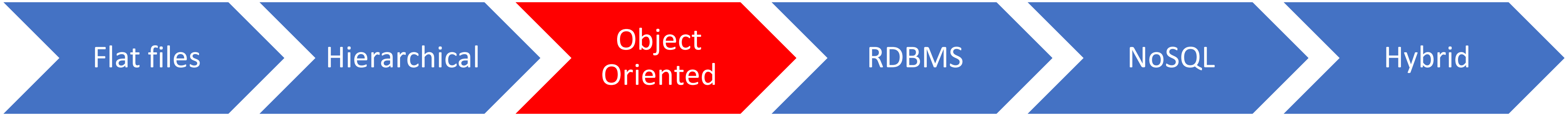
RDBMS

NoSQL

Hybrid

- Tree structure, examples: file system, Windows Registry
- IBM Information Management System (IMS)
- XML, XAML
- Used in mainframe era.
- Rigid structure.
- Only *One-to-many* relationship.
- Traversing very easy, moving a node difficult

DBMS (Database Management System)



Flat files

Hierarchical

Object Oriented

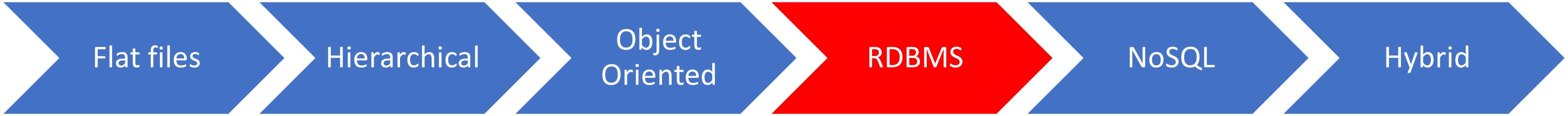
RDBMS

NoSQL

Hybrid

- Hybrid relation + objects =>> tables of objects.
- Realm database for Android/IoS: classes used as schema definition, alternative for SQLite.
- Next: MongoDB Realm.

DBMS (Database Management System)



Flat files

Hierarchical

Object Oriented

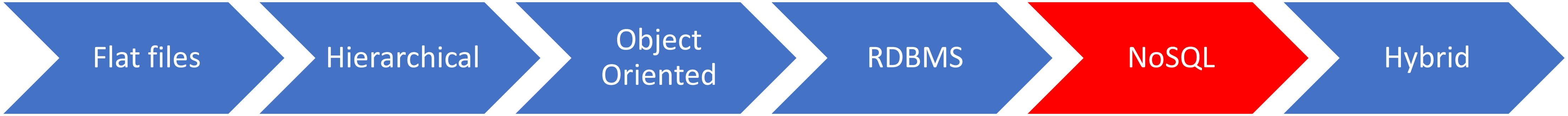
RDBMS

NoSQL

Hybrid

- Transaction oriented systems (financial transactions).
- ACID: Atomicity, consistency, isolation, durability.
- Suitable for structured data.

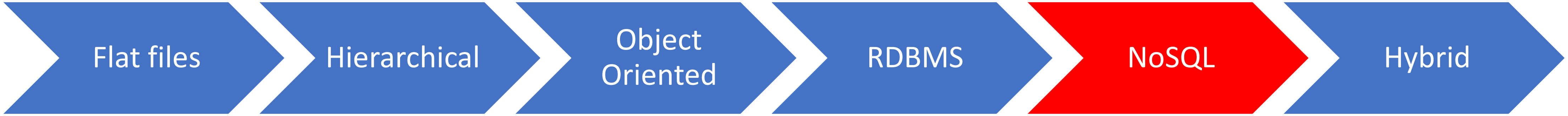
DBMS (Database Management System)



Flat files → Hierarchical → Object Oriented → RDBMS → NoSQL → Hybrid

- RDBMS hard to scale (only scale vertically, not horizontally).
- RDB Restrictive schemas =>> flexible structure.
- The state of the database can change.
- !!! availability, scalability, performance
- Sharding: distribute data on different servers

DBMS (Database Management System)



Flat files → Hierarchical → Object Oriented → RDBMS → NoSQL → Hybrid

- Cloud and bigdata.
- BASE (Basically Available, Soft state, Eventually consistent)
- Types:
  - key-value: Redis
  - Document: Mongo
  - Column: Apache Cassandra
  - Graph: Neo4j

Sql or NoSQL

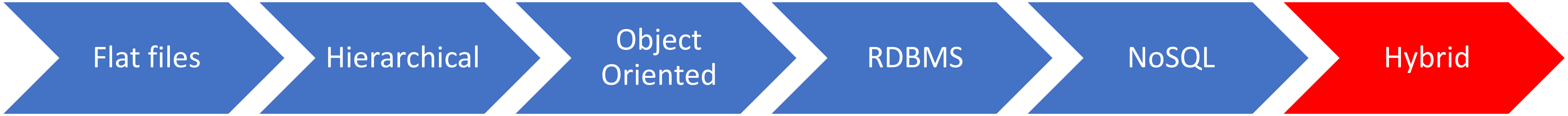
Relational

- Vertical scalability
- ACID
- pre-defined schema
- SQL language
- Normalized data

NoSql

- Horizontal scalabitily
- BASE
- Flexible schema
- No standard
- Collections, redundancy

DBMS (Database Management System)



Flat files

Hierarchical

Object Oriented

RDBMS

NoSQL

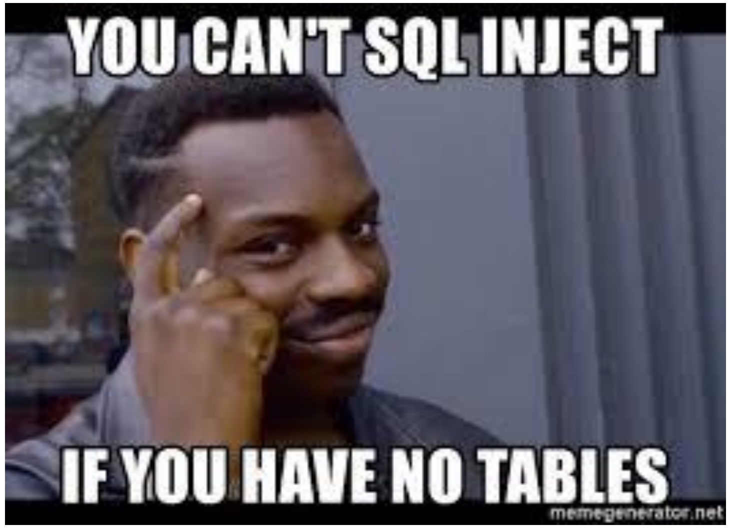
Hybrid

- Integration of Relational and NoSQL databases.
- Integration of in-memory DB and on-disk DB
- Altibase, Orient DB

Course roadmap

- Database design
- SQL
- NoSql
- … & other topics …

Course roadmap



ER diagram

- Visual representation of the ER model.
- Describes the logical structure of the (relational) database.
- Proposed by Chen in 1971.
- Easy to translate into relational tables.
- High-level design.
- Suitable for structured systems.

ER - Diagram

Entity Relationship model

ER - Diagram

- Visual representation of the ER conceptual data model.
- Describes the structure of the (relational) database.
- Not linked to the implementation or hardware.

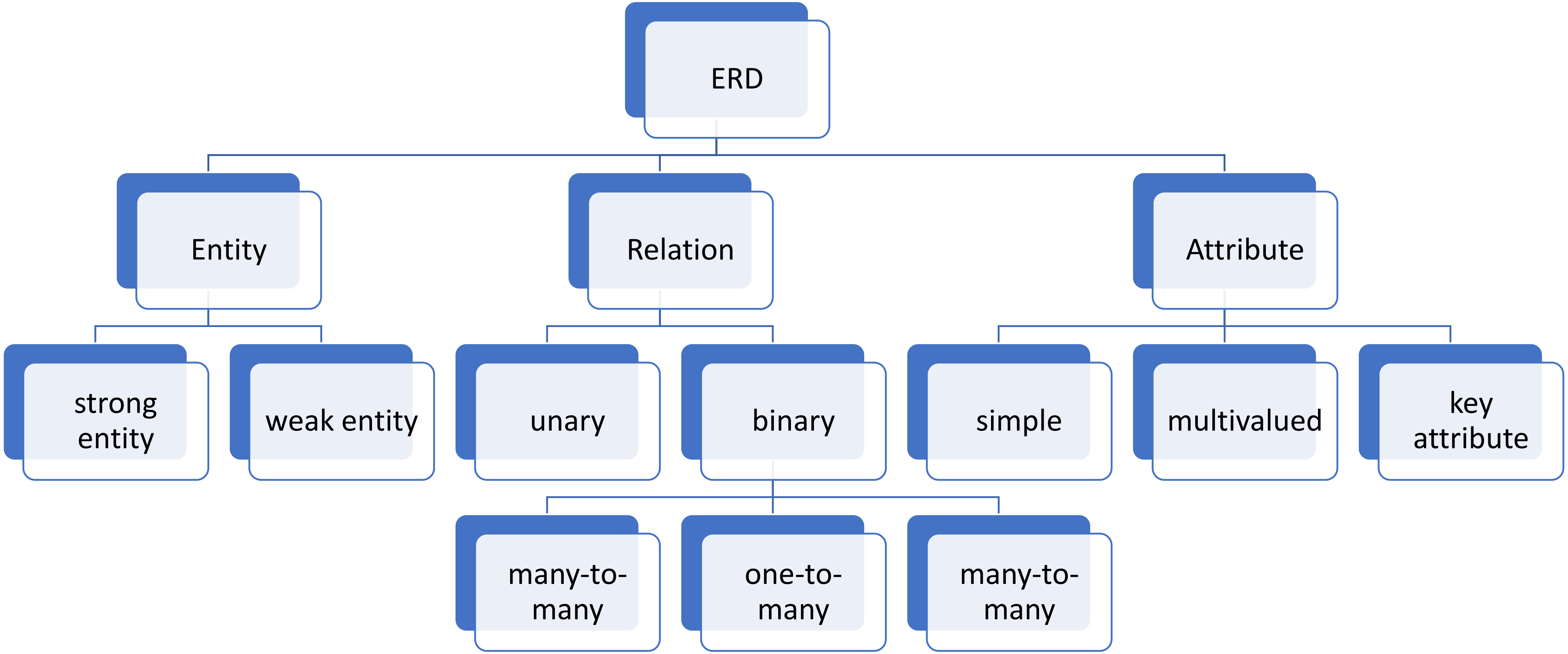
- Peter Chen developed ERDs in 1976.

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

ENTITY

person, place, activity, event, concept, real world object etc.

usually a noun

RELATION

ATTRIBUTE

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

ENTITY

person, place, activity, event, concept, real world object etc.

usually a noun

RELATION

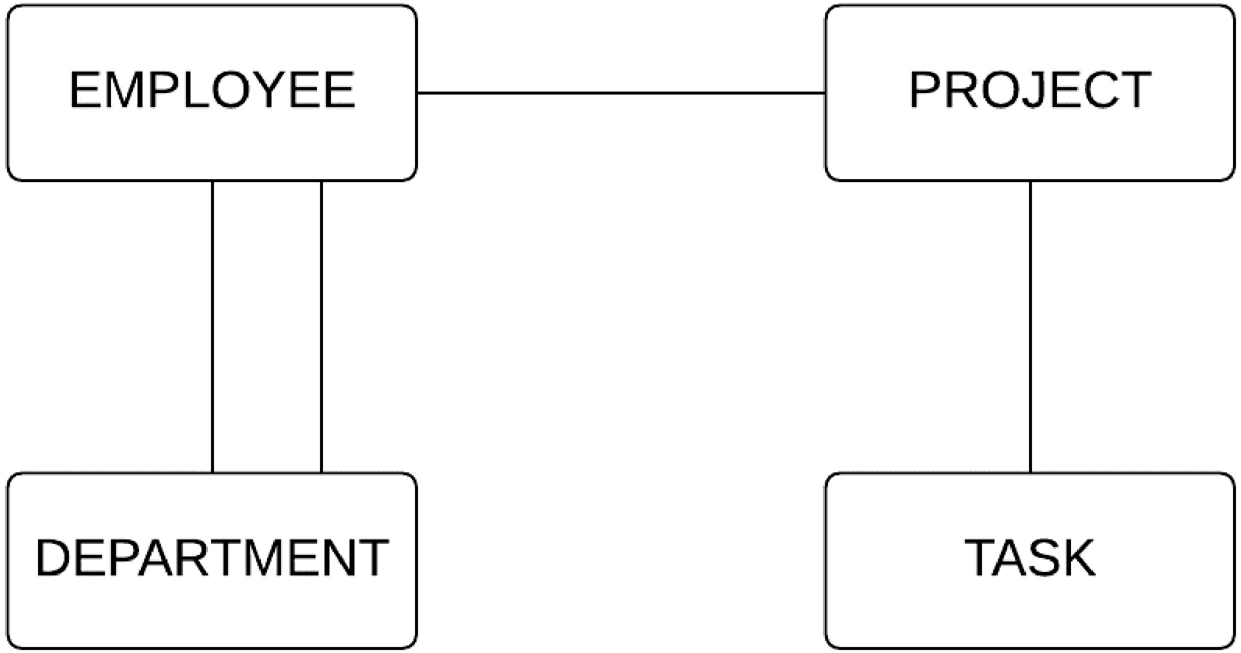
links entities (unary, binary, ternary).

usually a verb

ATTRIBUTE

describe entities or relations

Entities



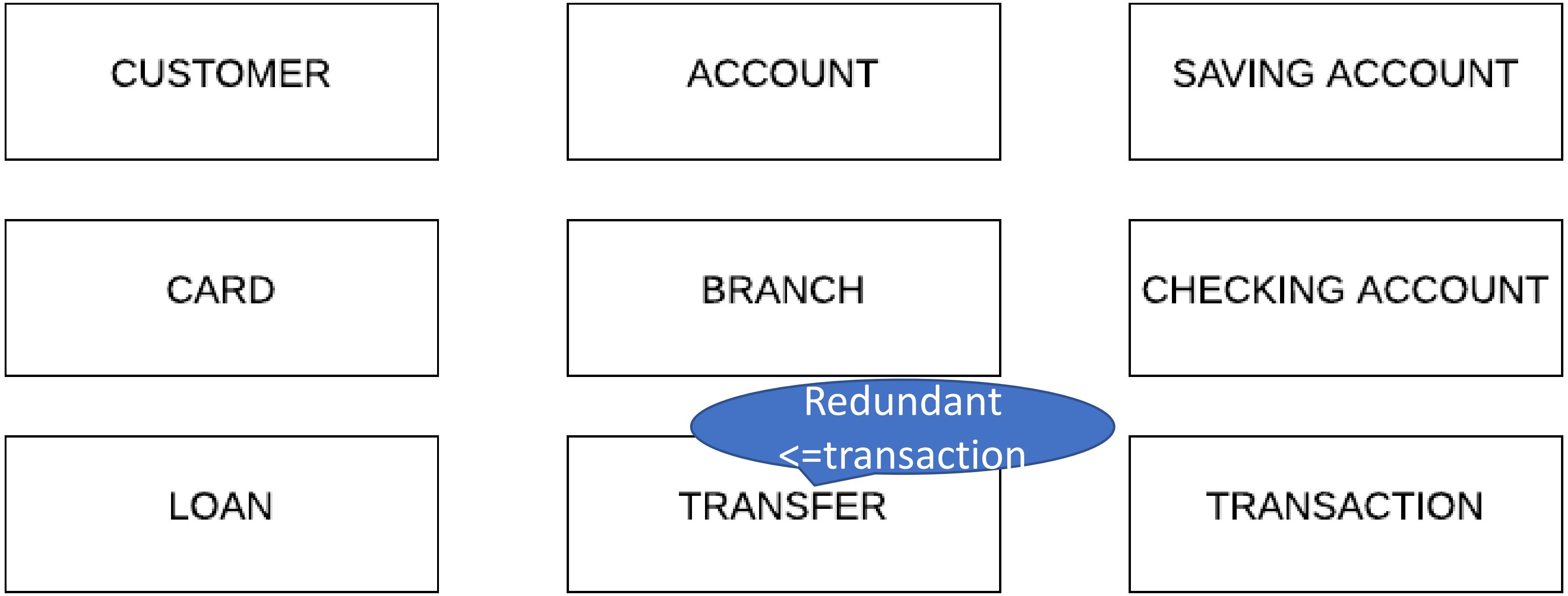
Banking (1)Entities

- 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.

Entities

- Unique names, uppercase characters
- Graphical representation: rectangles

- Relational database: entity ➔ table (line & columns)
- Primary key: attribute or group of attributes that uniquely identifies an entity instance



CUSTOMER

ACCOUNT

SAVING ACCOUNT

CARD

BRANCH

CHECKING ACCOUNT

LOAN

TRANSFER

TRANSACTION

Redundant <=transaction

Primary key

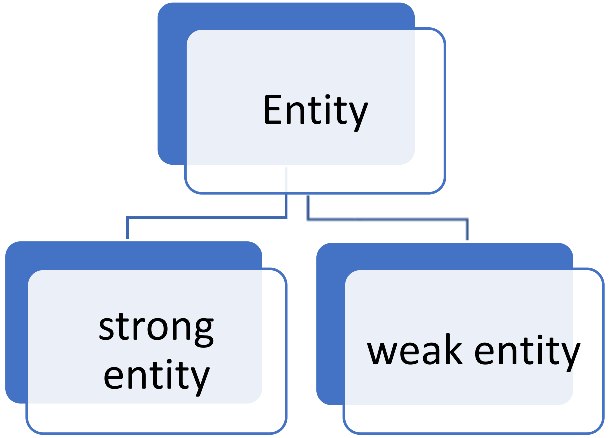
- Unique identifier
- Must be known at any moment
- Simple
- No ambiguities
- Immutable

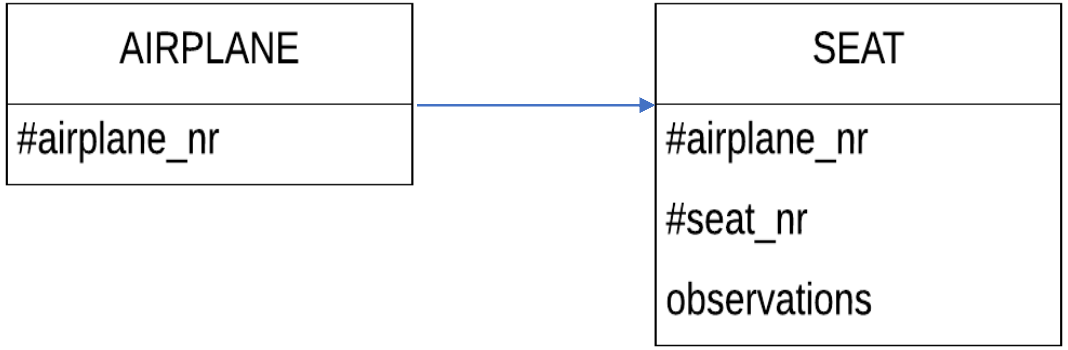
- Composed keys may be replaced with an artificial key.

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.

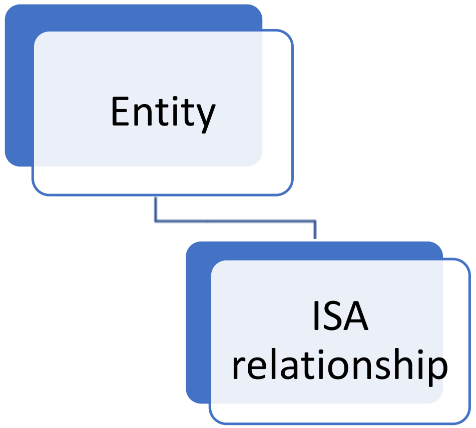
Entities

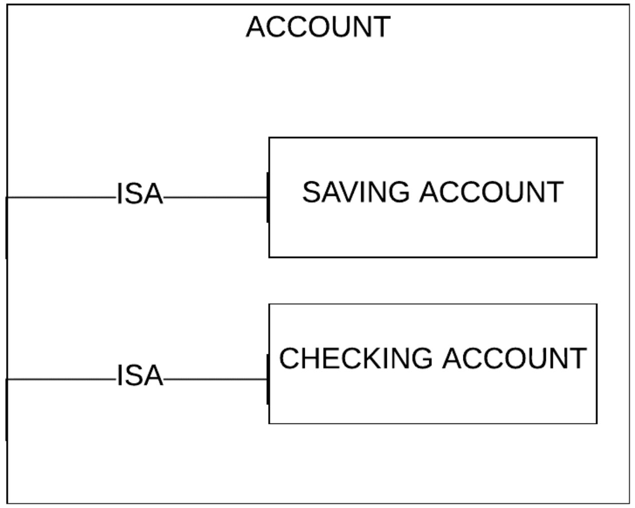




- Weak entity is an entity that depends on another entity.
- The primary key of a weak entity contains the primary key of the strong entity that it depends on + description/partial key.

Entities





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

Relationship

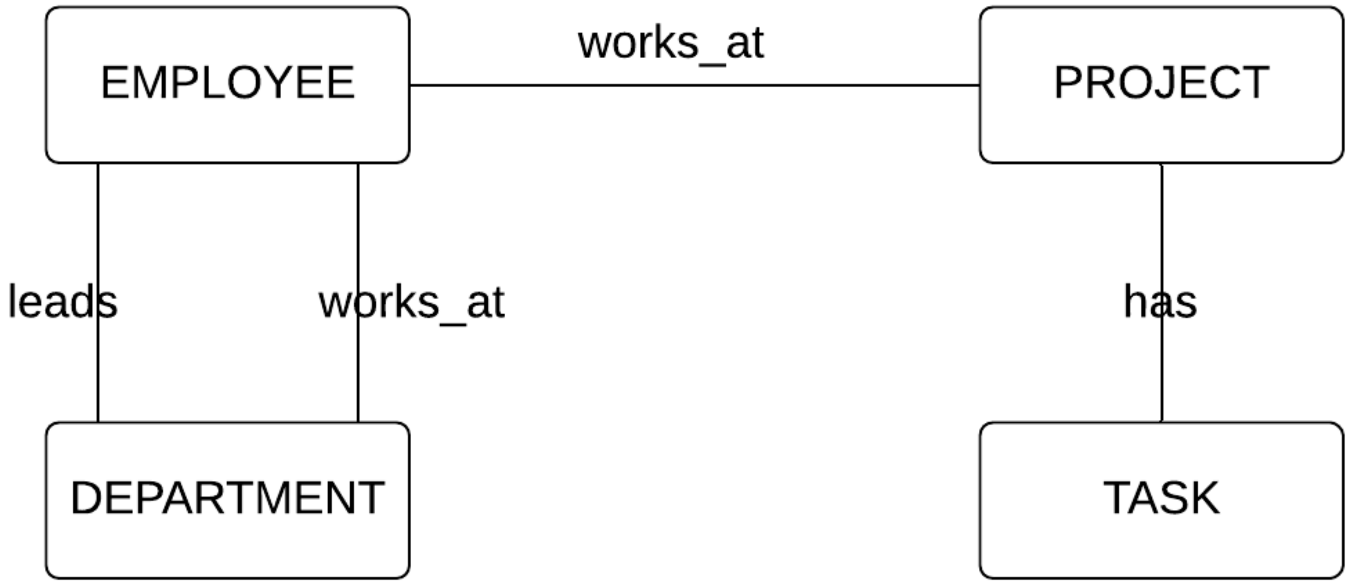
- Association between two or more entities (binary, ternary etc.)
- Relationship → column (foreign key) or table.
- Graphical representation: oriented arc.

- Two relationships with the same name link different entities.

- Cardinality defines the numerical attributes of the relationship between two entities: MANDATORY (min) OPTIONAL (max)

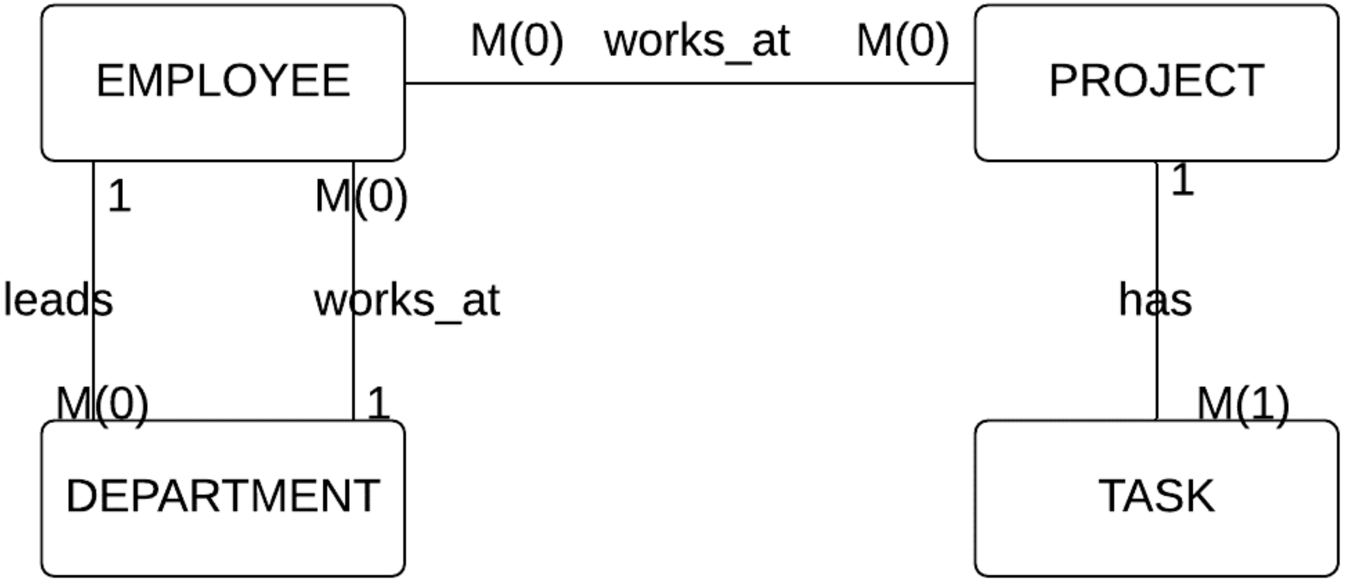
Relationship cardinality

- MANDATORY (must)
- OPTIONAL (may)



Relationship cardinality

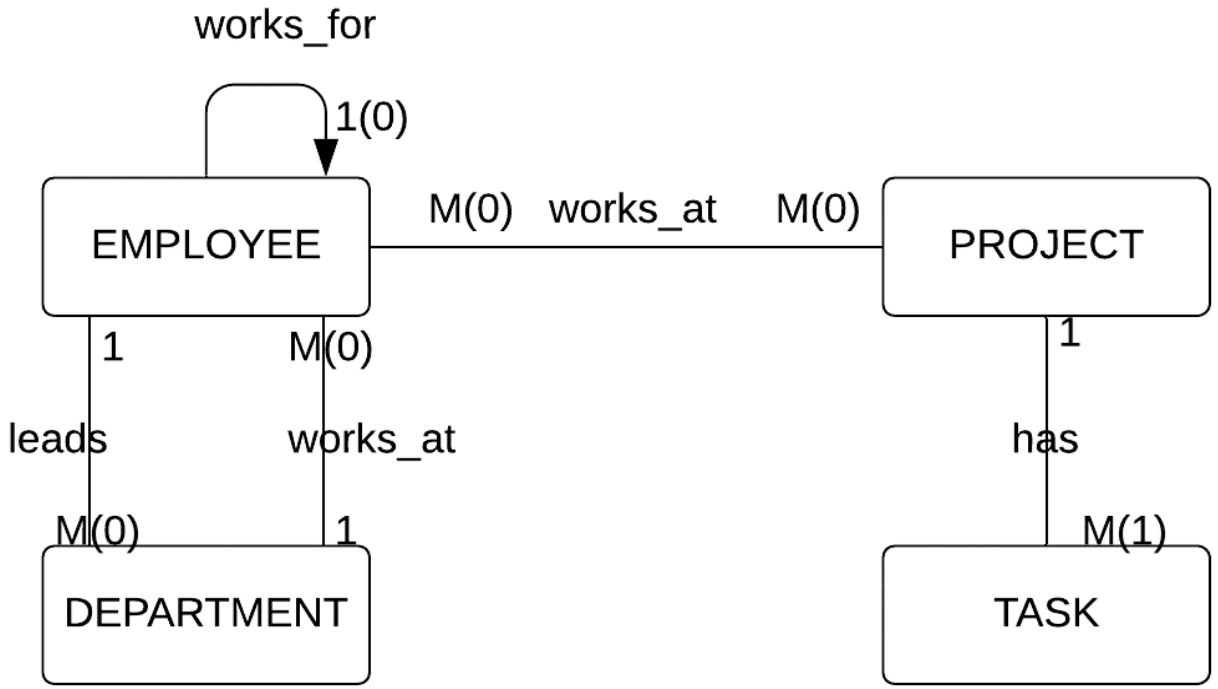
- MANDATORY (must)
- OPTIONAL (may)

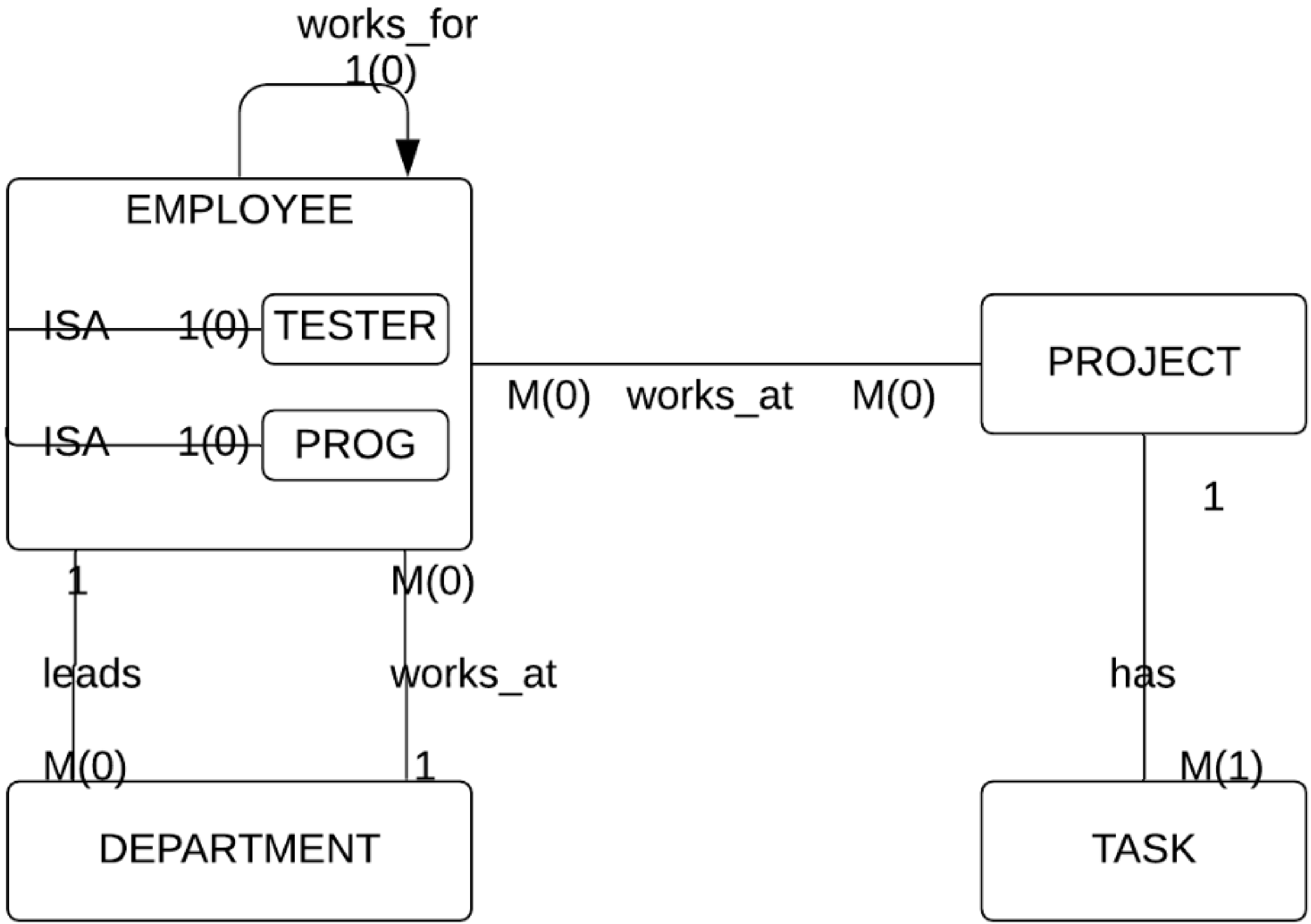


Relationship cardinality

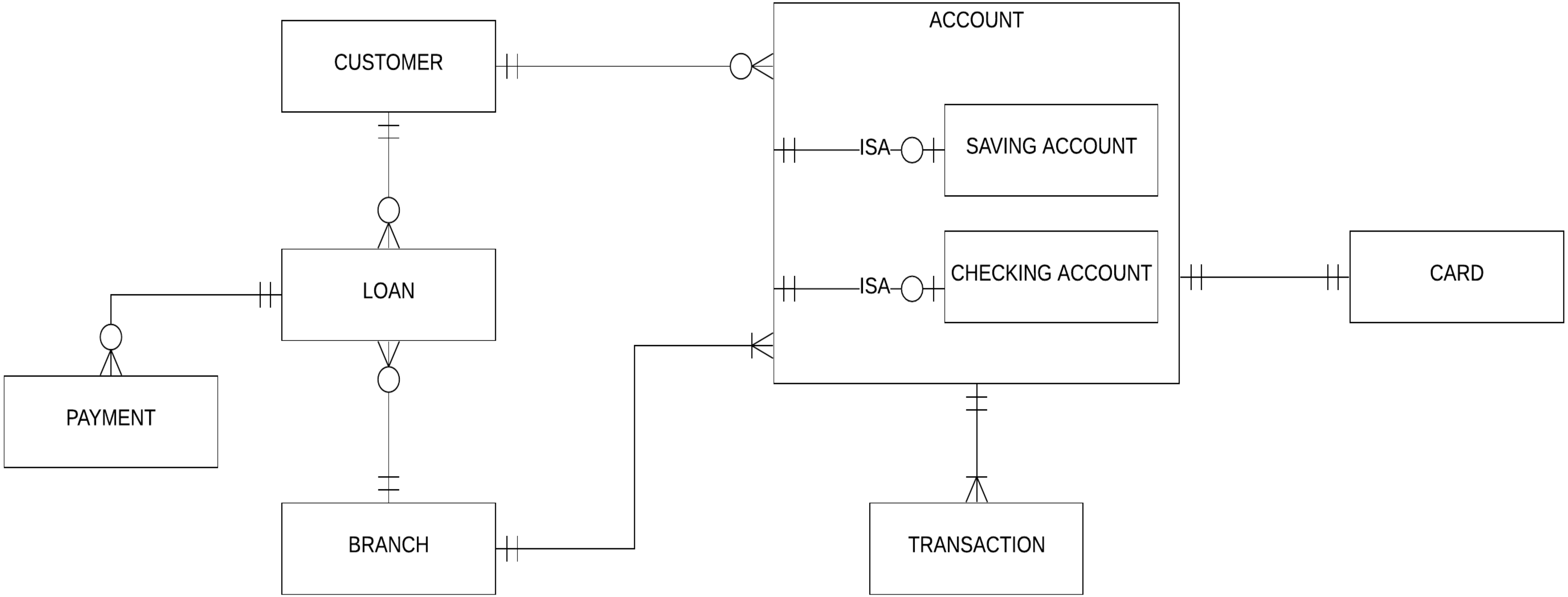
- Reflexive relationship

  unary relationship.



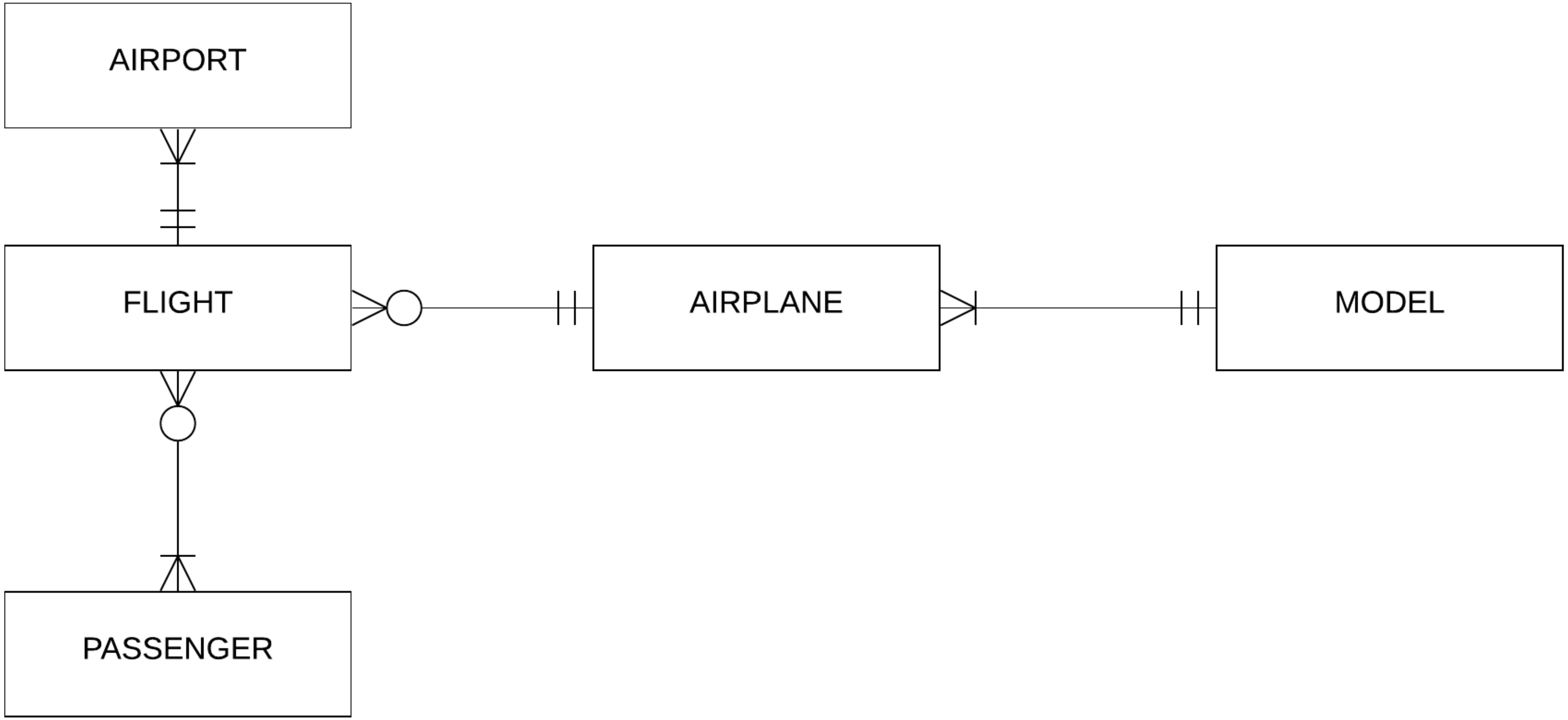

works_for
1(0)
EMPLOYEE
ISA
1(0)
TESTER
ISA
1(0)
PROG
M(0)
works_at
M(0)
PROJECT
1
1
M(0)
leads
works_at
has
M(0)
1
M(1)
DEPARTMENT
TASK

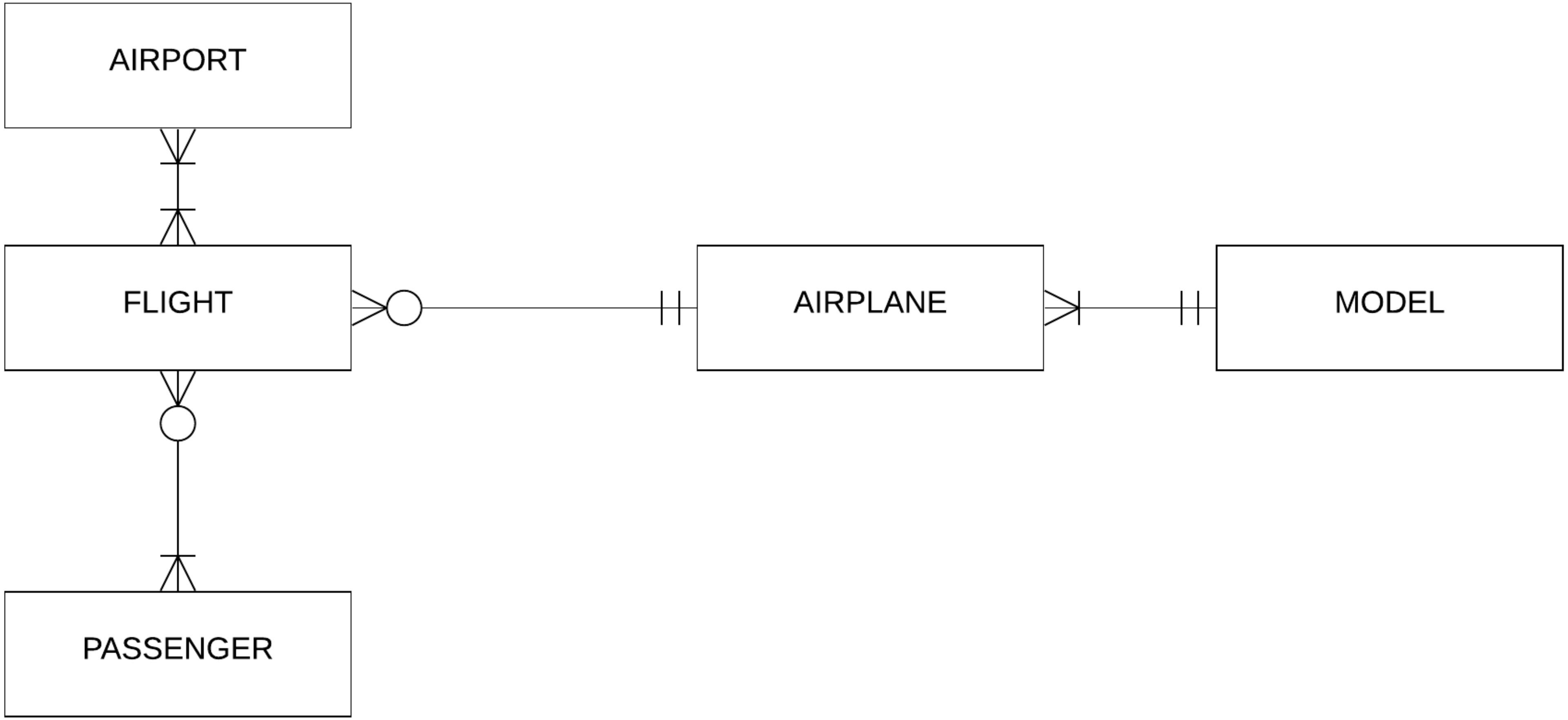
Banking (2)Relationships



Airline (1)Relationships

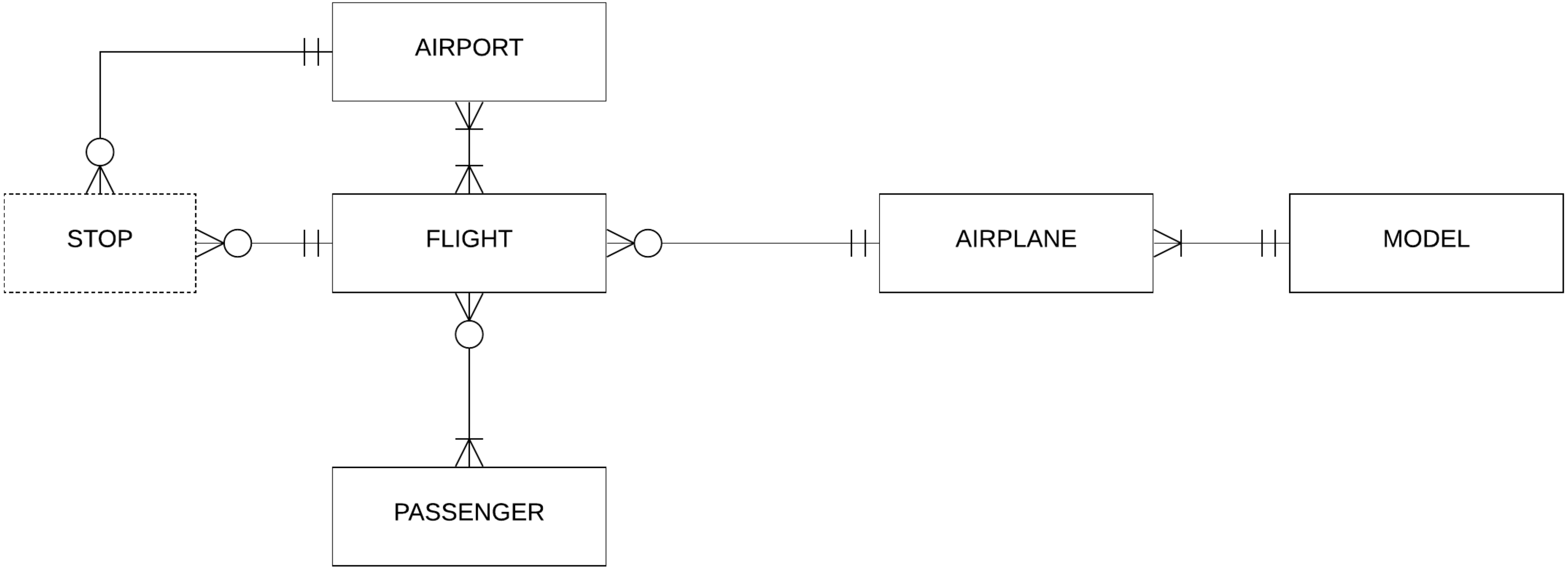
- The airline has one or more airplanes. An airplane has a model number, and capacity. Each flight is carried out by airplanes. An airplane is uniquely identified by its Registration_No and a flight is identified by its Flight_No. A passenger can book a ticket for a flight.


AIRPORT
FLIGHT
AIRPLANE
MODEL
PASSENGER


AIRPORT
FLIGHT
AIRPLANE
MODEL
PASSENGER

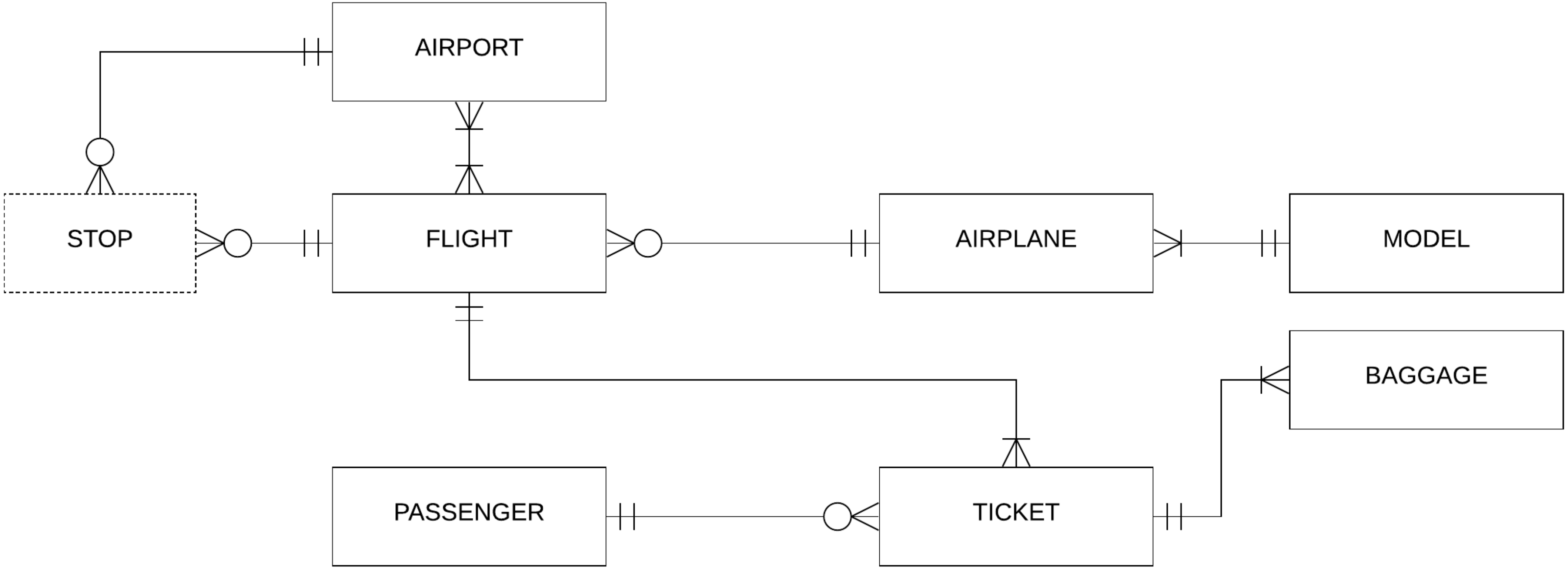
Airline (1)Relationships

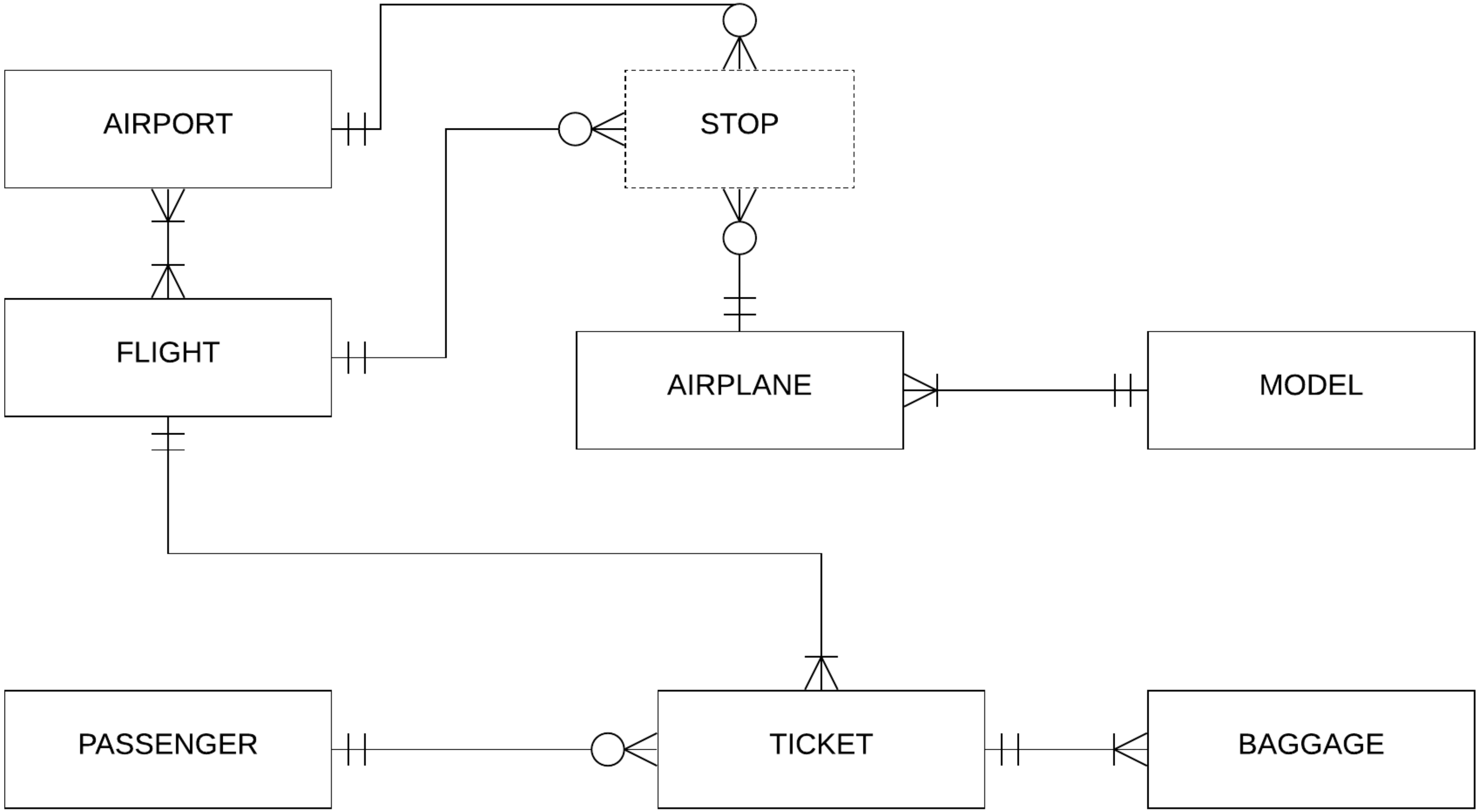
- The airline has one or more airplanes. An airplane has a model number, and capacity. Each flight is carried out by airplanes. An airplane is uniquely identified by its Registration_No and a flight is identified by its Flight_No. A passenger can book a ticket for a flight. A flight may have one or more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


AIRPORT
STOP
FLIGHT
AIRPLANE
MODEL
PASSENGER

Airline (1)Relationships

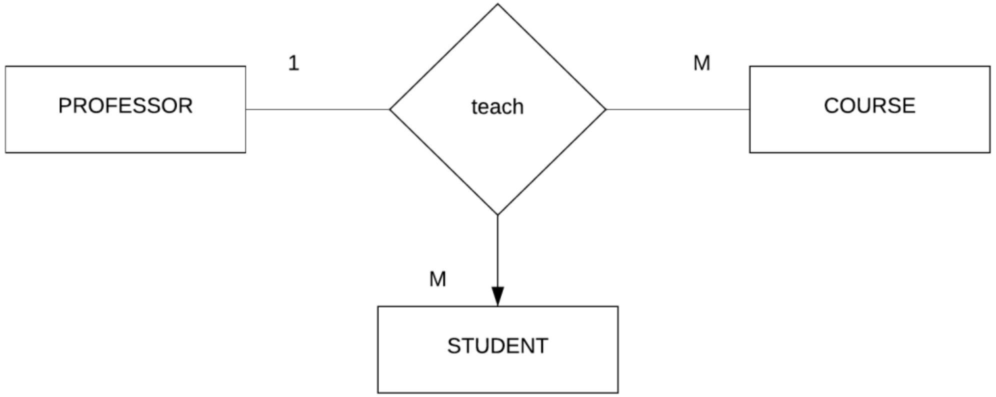
- The airline has one or more airplanes. An airplane has a model number, and capacity. Each flight is carried out by airplanes. An airplane is uniquely identified by its Registration_No and a flight is identified by its Flight_No. A passenger can book a ticket for a flight. A flight may have one or more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


AIRPORT
STOP
FLIGHT
AIRPLANE
MODEL
BAGGAGE
PASSENGER
TICKET


AIRPORT
STOP
FLIGHT
AIRPLANE
MODEL
PASSENGER
TICKET
BAGGAGE

Ternary relationships

- Relationship binding simultaneously 3 entities.



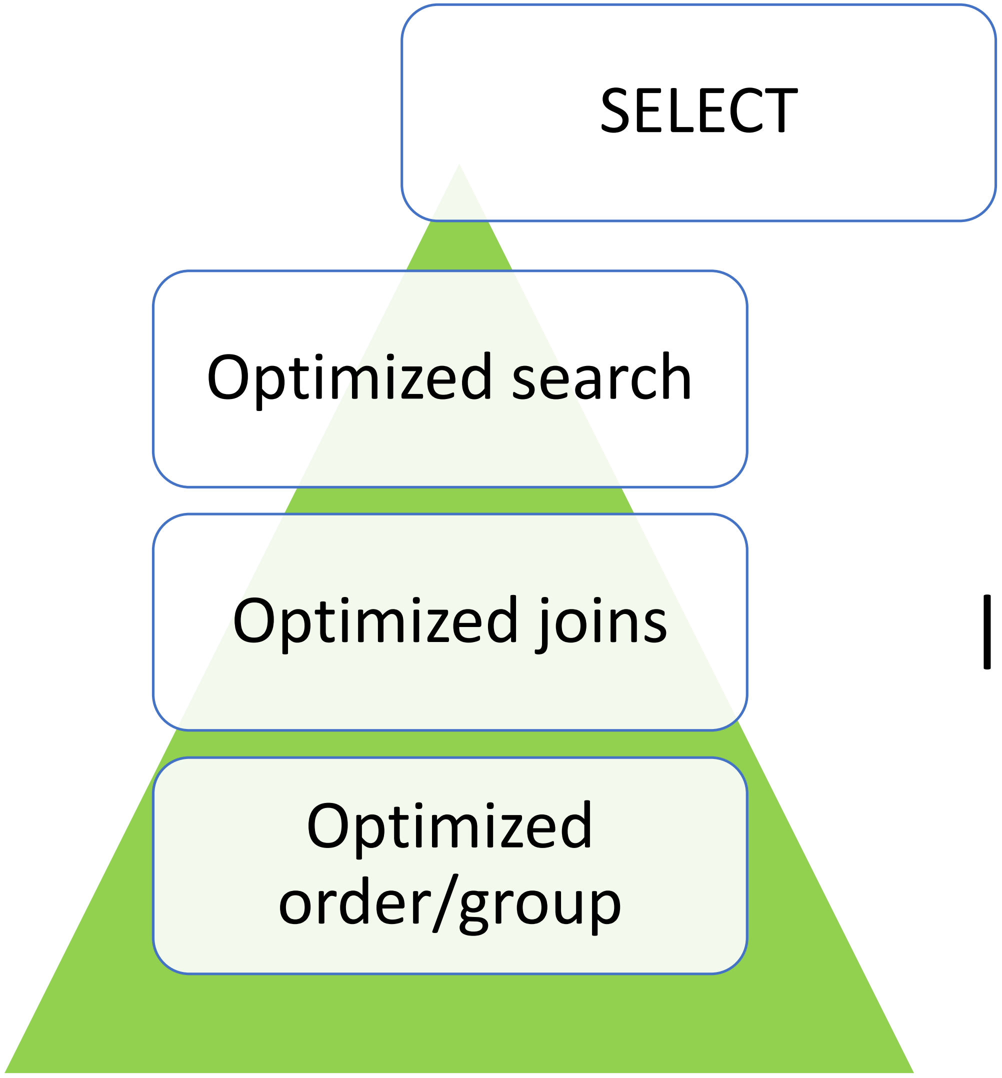
Indexes

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



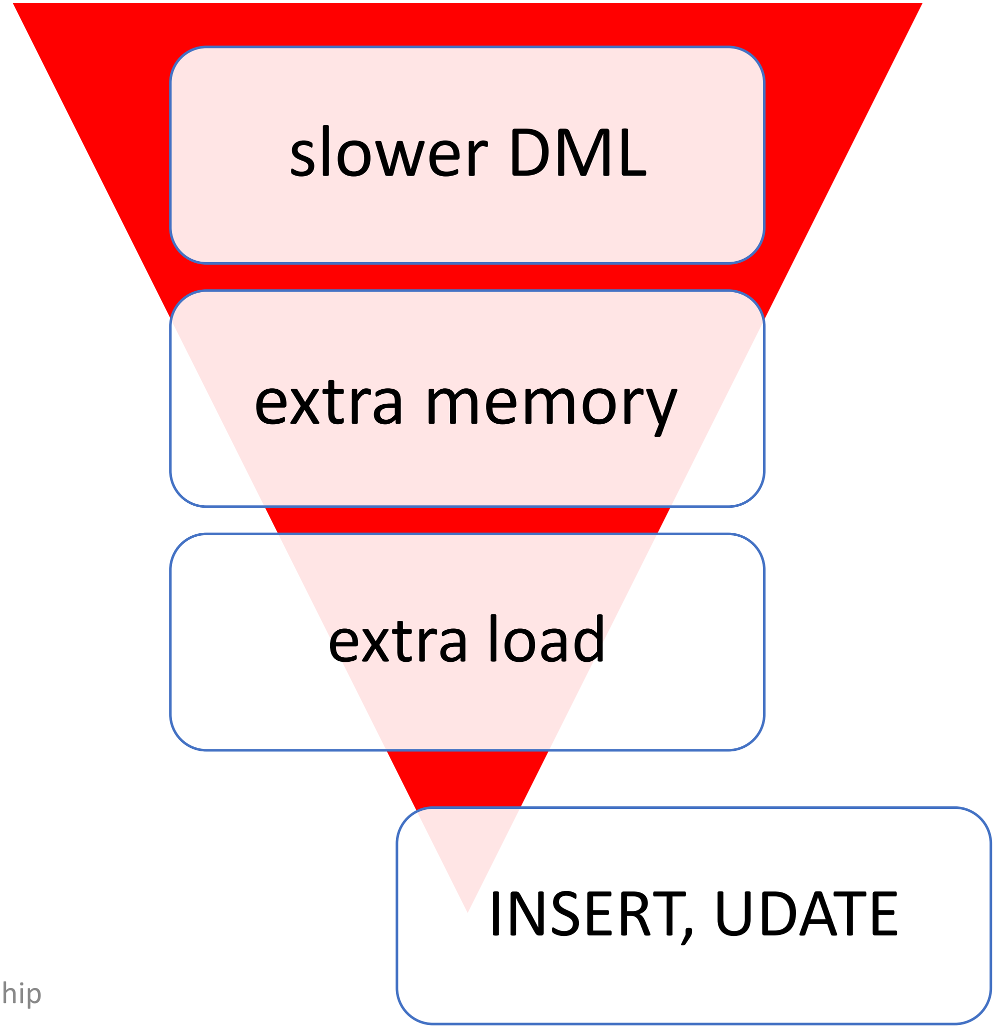
SELECT

Optimized search

Optimized joins

Optimized order/group

Index



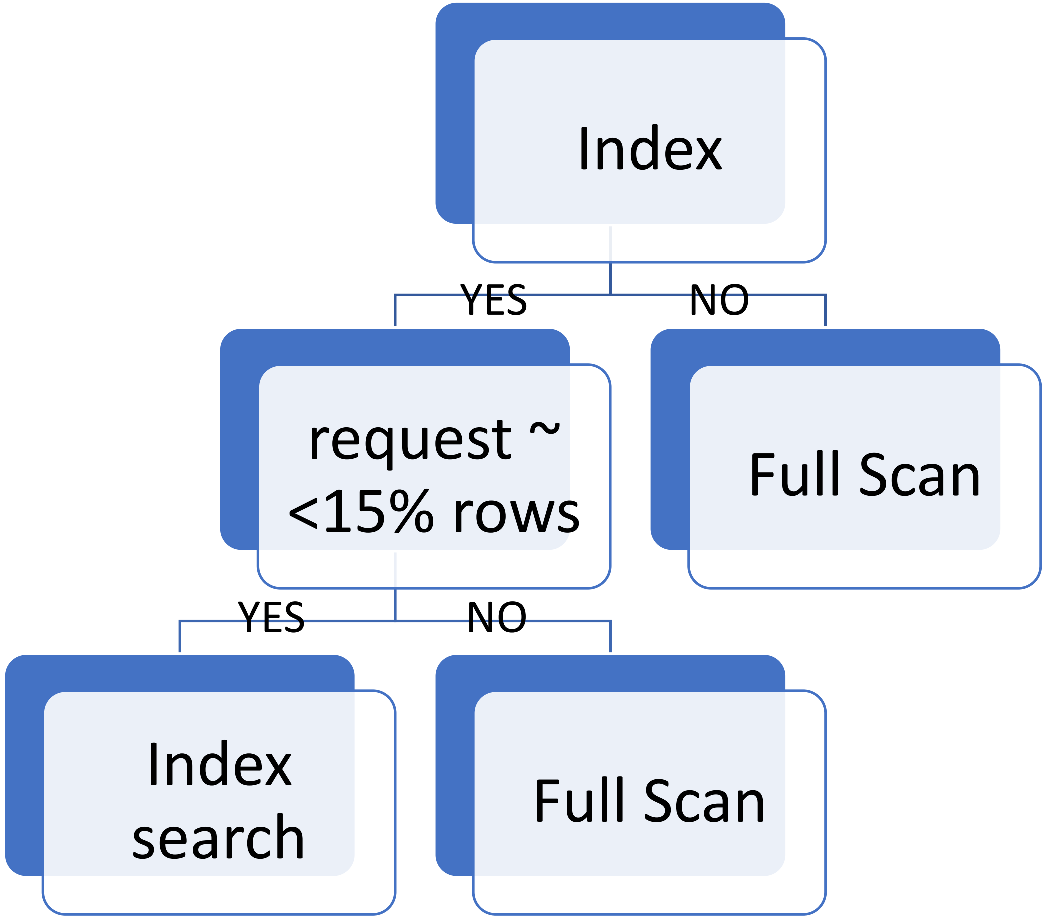
slower DML

extra memory

extra load

INSERT, UDATE

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';
```

Primay 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';
```

Index types

Clustered index (SqlServer, MySql)

- Defines the order in which data is physically stored in a table. (index on column semester)
- Only one clustered index on a table (data can be stored in only one order)
- A cluster index is created automatically when a primary key is defined.
- No second data structure for the table

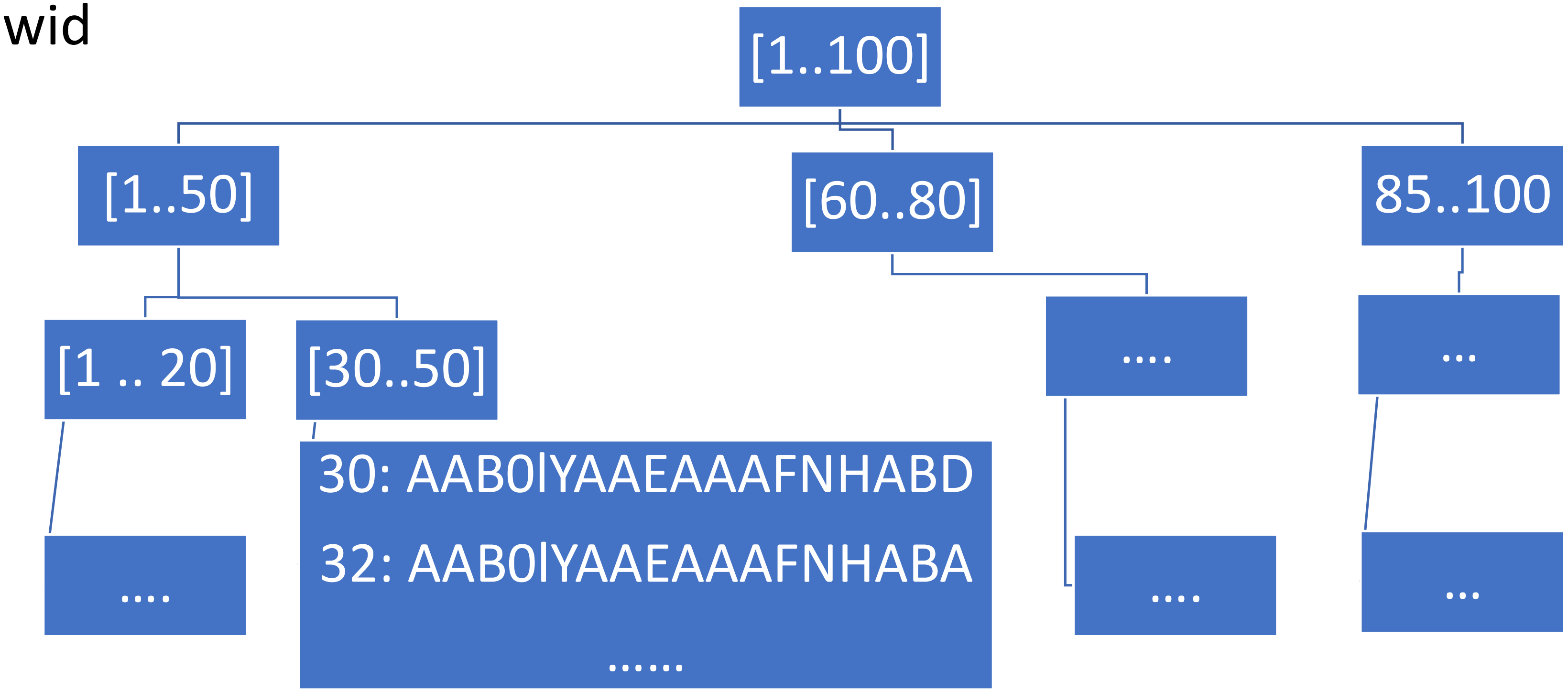
- Oracle: IOT index organized tables. Table is stored in a B-tree structure. (key and non-keys column are stored in leafs)

B – Tree

- B -- Balanced tree.
- Default index type in Oracle.
- Two types of nodes: branch blocks and leaf blocks.
- Branch blocks pointers to lower levels.
- Leaf blocks contain rowids/physical address.
- The number of blocks traversed in order to reach a leaf block is the same for each leaf block.

B – Tree

- create index idx_emp_id on employees(employee_id).
    - Devide employee_id values in sorted ranges.
    - Leafs nodes store rowid



Reverse index

- B – tree where keys are in reverse order. Key 4573 is stored 3754.
- Optimized insert operations.
- Key 4573 will be stored in the same block with key 9573

  while 4574 will be stored in a different block.

Bitmap index

- Used for columns with limited number of distinct values.
- Example: language proficiency levels (en)

emp_id	en	fr
1	A1	B1
2	A2	B2
3	C1	A1
4	A1	B1
5	A1	

row_id	A1	A2	B1	B2	C1	C2
AAB0lYAAEAAAFNHABD	1	0	0	0	0	0
AAB0lYAAEAAAFNHABV	0	1	0	0	0	0
AAB0lYAAEAAAFNHABX	0	0	0	0	1	0
AAB0lYAAEAAAFNHAAv	1	0	0	0	0	0
AAB0lYAAEAAAFNHAAV	1	0	0	0	0	0