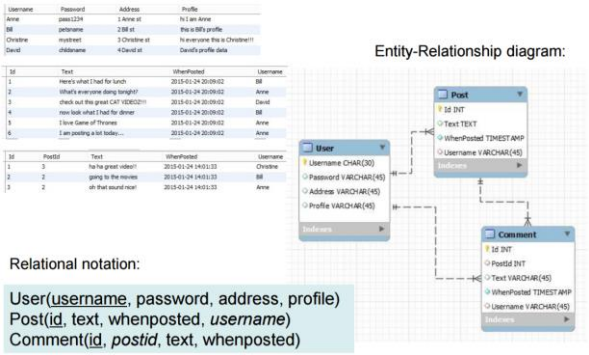
Database Systems and Information Modelling

Week 1

Design a database:

Maintaining contents of a database

- SELECT: read data from the table
- INSERT: new rows into the table
- DELETE: existing rows from the table
- UPDATE: existing row from the table



Entity-Relationship diagram:

Relational notation:

User(username, password, address, profile)
Post(id, text, whenposted, *username*)
Comment(id, *postid*, text, whenposted)

3 types of database:

Table form

Entity Relationship Diagram

- Use Data Definition Language (**DDL**) to manipulate the structure of the tables
- CREATE, DROP (delete a table), ALTER (add column), RENAME

Relational Notation

Database lifecycle

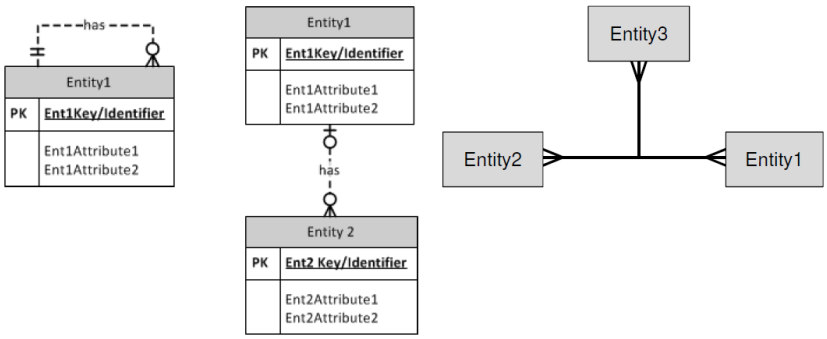
- Design the database
  - Data modelling, E-R diagrams
- Implement the database
  - Data definition language (**DDL**)
    - Create
    - Drop
    - Alter
    - Rename
- Data access / programming
  - Data manipulation language (**DML**) - CRUD
    - **C**reate (Insert)
    - **R**ead (Select)
    - **U**pdate
    - **D**elete
- Database administration
  - Data control language (**DCL**)
    - Grant
    - Revoke

Noun-Verb analysis:

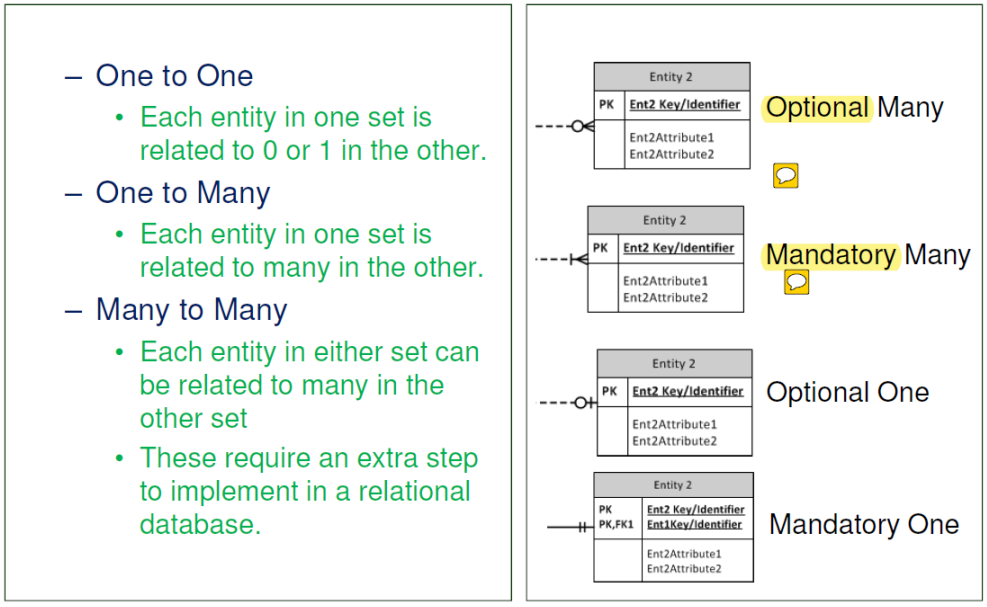
- **Nouns**: Tables (rows)
- **Verbs:** Describe the entity (relationship between nouns)
  - One employee to one department OR multiple departments
- **Adjectives**

Relationship Degree

Unary Binary Ternary



Relationship Cardinality (Ignore dotted lines)



Optional Many: Can have a customer that has not yet placed an order.

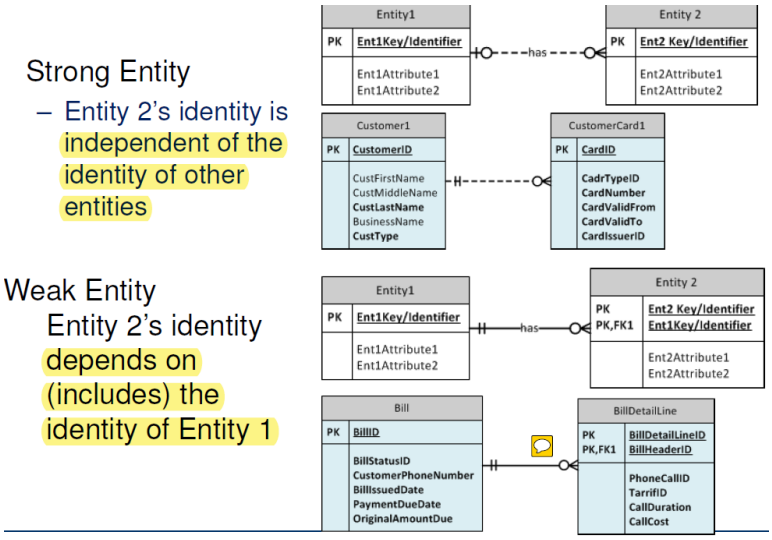
- Some read as 0 to many or optional to many
- Could have several or no orders

Mandatory Many: Does not have any customers **UNLESS** there is a purchase

- This depends the BUSINESS RULE: Do people count as customers if they haven't made a purchase?

Strong/Weak entities

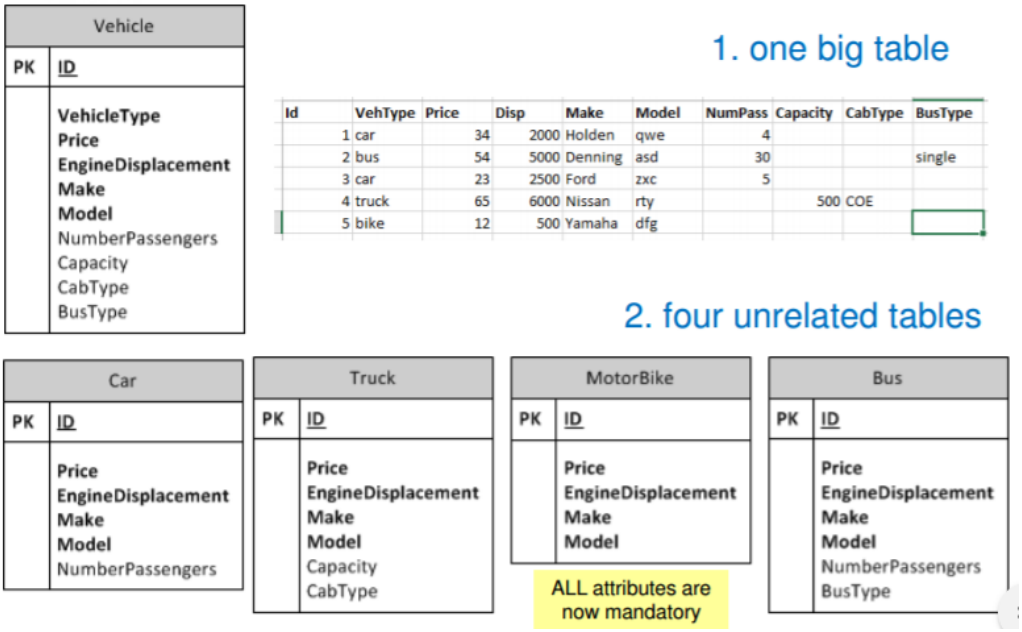
- Weak: PK of entity 2 is also PK of entity 1 (THUS replies on each other)



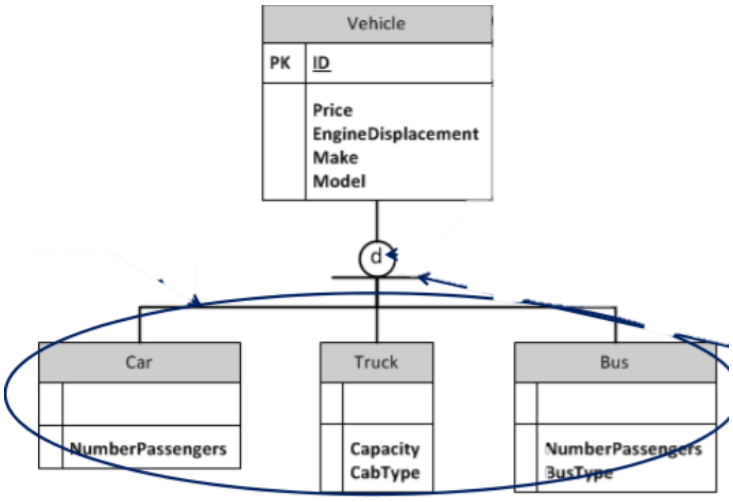
Week 5:

Subtypes and supertypes

Without subtyping, often will get a lot of repeated columns.



With subtyping:



- Each of the subtypes inherits all of the attributes of the supertype (Vehicle)
- Motorbike disappears as a Vehicle could either be a Car, Truck, Bus or none of them
- 'd' means disjoint: ONLY be one of these
- Single line under the 'd' says needn't be any

SUBTYPE types:

- Disjointness Constraints: 'd' or an 'o'
  - 'd' = disjoint (can only be one of those)
  - 'o' = overlapping (can be more than one of these)
- Completeness Constraints – specifies whether an instance of a supertype must also be an instance of a subtype (OR partial)
  - Double line: entity of type subtype1 MUST also be a subtype
  - Single line: Doesn't need to be one of the subtypes

An entity CAN have relationship with 1 of the subtypes

IFNULL ()

- Can convert a null to a 0 (can be useful in calculations)
- SELECT 1 + IFNULL (wagevalue, 0)
- Gives 1 + 0 for null fields and 1 + wage value for non-null fields
- Failure to do this results in a null answer for values where wage value is NULL

UPPER () / LOWER ()

- Change string to upper / lower case

LEFT ()

- Returns the leftmost X characters from the string
- SELECT LEFT ("This is a test", 6)
- Gives "This I"

RIGHT ()

- SELECT RIGHT ("This is a test", 6)
- Gives "a test"

More on INSERT:

- Insert records from another table
    - INSERT INTO NewEmployee SELECT * FROM Employee;
        - Emplyoee must already exist
- Insert multiple rows:
    - INSERT INTO EMPLOYEE VALUES
      (DEFAULT, "A", "A's Addr", "2012-02-02", NULL, "S"),
      (…);
    - INSERT INTO Employee
      (Name, Address, Datehired, EmployeeType)
      VALUES
      ("D", "D's Addr", "2012-02-02", "C"),
      (…);

More on UPDATE:

```
UPDATE Hourly
    SET HourlyRate = HourlyRate * 1.10;
```

- Increase salaries greater than $100K by 10% and rest 5%

```
UPDATE Salaried
    SET AnnualSalary = AnnualSalary * 1.05
    WHERE AnnualSalary <= 100000;
UPDATE Salaried
    SET AnnualSalary = AnnualSalary * 1.10
    WHERE AnnualSalary > 100000;
```

BUT: This method is very slow as we are rerunning and testing every row twice. Change to:

```
UPDATE Salaried
    SET AnnualSalary =
        CASE
            WHERE AnnualSalary <= 100000
            THEN AnnualSalary * 1.05
            ELSE AnnualSalary * 1.10
        END;
```