

E-R model, Relational model, SQL

Hogeschool Rotterdam
Rotterdam, Netherlands

Lecture topics

- E-R model.
- Relational model.
- SQL, and examples.

Overview

- Highest level of database modelling.
- Model the conceptual aspect of the database.
- Far from the physical representation in the DBMS.

Entity

- Anything which can exist on its own on the database
- Consider a database for a space shooter game
- Starships, asteroids are entities, they have a meaning on their own

Attributes

- They model characteristics of the entity.
- **Starship:** velocity, shield, armour, weapon, [...]
- **Asteroid:** velocity, mass, integrity, [...]

Relations

- They describe the associations among entities (two or more).
- They have a cardinality: number of participants for each side.

Relations - 1 : 1

- Entity modelling a pilot and one modelling a starship.
- Related by “drives”.
- The cardinality is 1:1 : one pilot drives at most one starship, and one starship can contain only one pilot.

Relations - 1 : N

- Entity modelling a starship and one modelling a weapon.
- Related by “mounted”
- The cardinality is 1:N : a weapon can be mounted only on one starship, but a starship can mount more than one weapon.

Relations - N : M

- Entity modelling a starship and one modelling an asteroid.
- Related by “collides with”
- The cardinality is N : M : several starships can collide with several asteroids.

Keys

- A way to uniquely identify an entity.
- A key is a set of attributes that have unique values among entities.
- **Starship:** Serial number.

Weak entities

- Entities which do not have a key attribute.
- **Asteroids:** There can be two asteroids with the same position, same mass, velocity, etc.

Overview

- Halfway between a conceptual model and the physical model.
- Contain an abstraction of physical elements.
- Can be easily mapped to a physical implementation in a DBMS.
- There are mapping rules from E-R model to the relational model.

Relation

- A relation is a collection of tuples.
- Each element of a tuple is a value taken from an attribute set.
- Each attribute set is identified by a name

Ship				
<u>Serial</u>	Name	Shields	Armour	Integrity

(38258269, "Battlestar Galactica", 3000, 5000, 1.0)

Keys

- A *Primary key* is a set of attributes with unique values in each tuple.
- A *Candidate key* is the smallest set of attributes which form a superkey.

Example:

Primary key: (Serial, Name, Shield)

Candidate key: (Serial)

Keys

- A *Primary key* is the chosen key for a relation among all the candidate keys.
- A *Foreign key* is a set of attributes in one relation which is a primary key in another relation.

Example (Foreign key):

Mounts	
<u>ShipSerial</u>	<u>WeaponName</u>

Ship				
<u>Serial</u>	Name	Shields	Armour	Integrity

In the relation Mounts the attribute ShipSerial is a foreign key to Ship.

Overview

- Used to create relations (tables).
- Used to insert/modify/extract data from relations (tables).
- Declarative language (“What” not “How”).

Ship				
<u>Serial</u>	Name	Shields	Armour	Integrity

Select all ships from the game

Ship				
<u>Serial</u>	Name	Shields	Armour	Integrity

Select all ships from the game

```
SELECT *  
FROM Ships
```

Ship				
<u>Serial</u>	Name	Shields	Armour	Integrity

Select all ships in the game whose pilot is “William Adama”

Ship				
<u>Serial</u>	Name	Shields	Armour	Integrity

Select all ships in the game whose pilot is “William Adama”

```
SELECT *  
FROM Ships s  
WHERE s.Pilot = 'William_Adama'
```

Ship				
<u>Serial</u>	Name	Shields	Armour	Integrity

Find the name of the ships whose pilot is “Starbucks”

Ship				
<u>Serial</u>	Name	Shields	Armour	Integrity

Find the name of the ships whose pilot is “Starbucks”

```
SELECT s.Name  
FROM Ship s  
where s.Pilot = 'Starbucks'
```

Ship				
<u>Serial</u>	Name	Shields	Armour	Integrity

Mounts	
<u>ShipSerial</u>	<u>WeaponName</u>

Weapon		
<u>Name</u>	Damage	Type

**Find the serial of the ships mounting the weapon
“Stealthblade MKII”**

Ship				
<u>Serial</u>	Name	Shields	Armour	Integrity

Mounts	
<u>ShipSerial</u>	<u>WeaponName</u>

Weapon		
<u>Name</u>	Damage	Type

**Find the serial of the ships mounting the weapon
“Stealthblade MKII”**

```
SELECT s.Serial
FROM Ship s, Mounts m
WHERE s.Serial = m.ShipSerial AND
      m.WeaponName = 'StealthBlade MKII'
```


Ship				
<u>Serial</u>	Name	Shields	Armour	Integrity

Mounts	
<u>ShipSerial</u>	<u>WeaponName</u>

Weapon		
<u>Name</u>	Damage	Type

Find the name of all the weapons mounted in the ships flown by “Apollo”

Ship				
<u>Serial</u>	Name	Shields	Armour	Integrity

Mounts	
<u>ShipSerial</u>	<u>WeaponName</u>

Weapon		
<u>Name</u>	Damage	Type

Find the name of all the weapons mounted in the ships flown by “Apollo”

```
SELECT w.Name
FROM Ship s, Mounts m, Weapon w
WHERE s.Serial = m.ShipSerial AND
      m.WeaponName = w.Name AND
      s.Pilot = 'Apollo'
```

Ship				
<u>Serial</u>	Name	Shields	Armour	Integrity

Mounts	
<u>ShipSerial</u>	<u>WeaponName</u>

Weapon		
<u>Name</u>	Damage	Type

**Find the total damage output of the ships flown by
“Athena”**

Ship				
<u>Serial</u>	Name	Shields	Armour	Integrity

Mounts	
<u>ShipSerial</u>	<u>WeaponName</u>

Weapon		
<u>Name</u>	Damage	Type

**Find the total damage output of the ships flown by
"Athena"**

```
SELECT SUM(w.Damage) AS Damage
FROM Ship s, Mounts m, Weapon w
WHERE s.Serial = m.ShipSerial AND
      m.WeaponName = w.Name AND
      s.Pilot = 'Athena'
```

Ship				
<u>Serial</u>	Name	Shields	Armour	Integrity

Mounts	
<u>ShipSerial</u>	<u>WeaponName</u>

Weapon		
<u>Name</u>	Damage	Type

Count all the ships having more than 3 weapons

Ship				
<u>Serial</u>	Name	Shields	Armour	Integrity

Mounts	
<u>ShipSerial</u>	<u>WeaponName</u>

Weapon		
<u>Name</u>	Damage	Type

Count all the ships having more than 3 weapons

```
SELECT COUNT(*)
FROM (
  SELECT COUNT(*) AS ShipCount
  FROM Ship s, Mounts m, Weapon w
  WHERE s.Serial = m.ShipSerial AND
        m.WeaponName = w.Name
  GROUP BY s.Serial
  HAVING COUNT(*) > 3)
```