The Hobbit: An Unexpected Database

Designed By: Megan Makini

Executive Summary

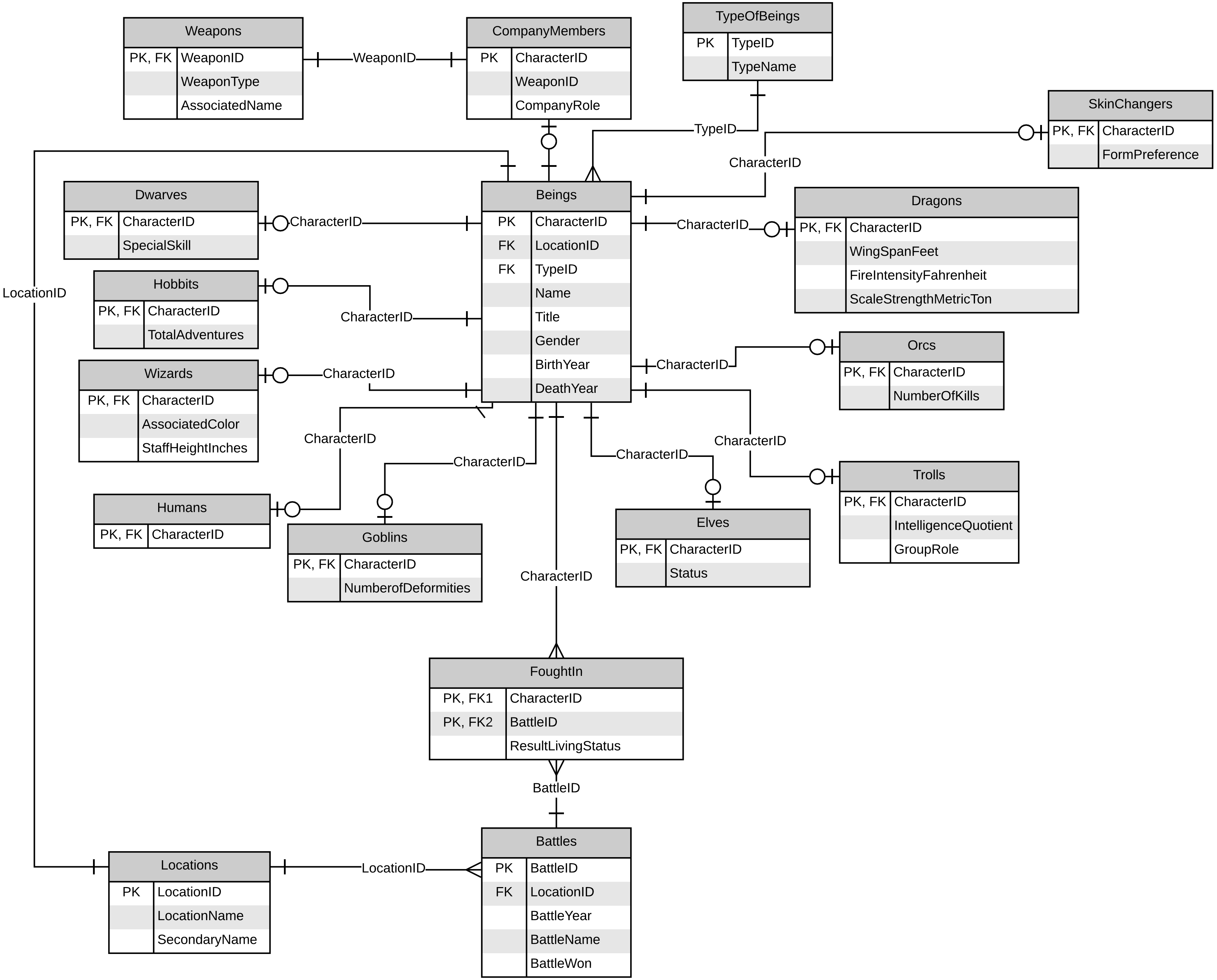
This document outlines the design and implementation of a database containing data on the beings, locations, and conflicts of Middle Earth from the years of 2941 and 2942 of the Third Age. To clarify, the information of the beings, locations, and conflicts predate events associated with The Lord of the Rings.

Beginning with the ER diagram, further discussion of each table will be presented with its individual SQL create statement and sample data. The discussion will then progress to views, reports, stored procedures, and triggers included within this database. Accompanying the discussion will be sample outputs to display the usefulness of each implementation. Lastly, the roles and security privileges, leading to notes on implementation and discussion of issues and future improvements.

Objective

This database and its implementation is intended to be used by Bilbo Baggins as he develops his autobiography, “There and Back Again,” intended for his second cousin Frodo Baggins. Because Bilbo does not commence his written work till many years later, the goal of this normalized database is to enforce the accuracy of encountered events, beings, and places. This database will allow Bilbo to efficiently access information from queries as his old age takes a toll on his mind. The general use of this database is intended to be used in a historical context, not requiring much updating, except for the death dates of certain beings.

ER Diägräm



Weapons	
PK, FK	WeaponID
	WeaponType
	AssociatedName

CompanyMembers	
PK	CharacterID
	WeaponID
	CompanyRole

TypeOfBeings	
PK	TypeID
	TypeName

SkinChangers	
PK, FK	CharacterID
	FormPreference

Dwarves	
PK, FK	CharacterID
	SpecialSkill

Beings	
PK	CharacterID
FK	LocationID
FK	TypeID
	Name
	Title
	Gender
	BirthYear
	DeathYear

Dragons	
PK, FK	CharacterID
	WingSpanFeet
	FireIntensityFahrenheit
	ScaleStrengthMetricTon

Hobbits	
PK, FK	CharacterID
	TotalAdventures

Wizards	
PK, FK	CharacterID
	AssociatedColor
	StaffHeightInches

Orcs	
PK, FK	CharacterID
	NumberOfKills

Trolls	
PK, FK	CharacterID
	IntelligenceQuotient
	GroupRole

Humans	
PK, FK	CharacterID

Goblins	
PK, FK	CharacterID
	NumberofDeformities

Elves	
PK, FK	CharacterID
	Status

FoughtIn	
PK, FK1	CharacterID
PK, FK2	BattleID
	ResultLivingStatus

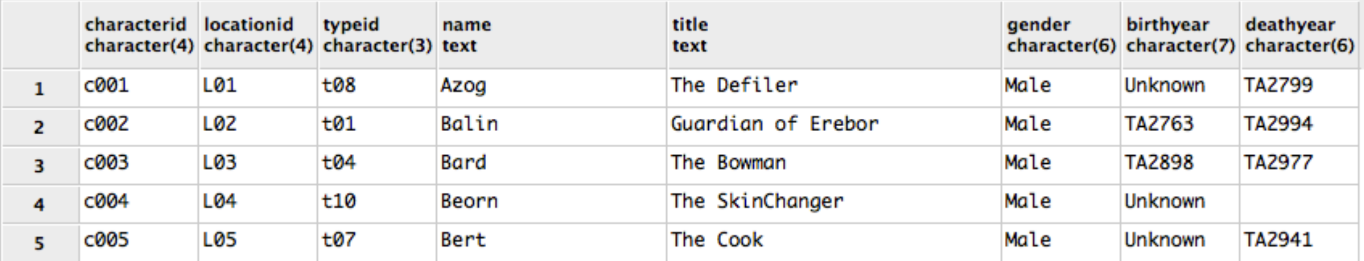
Locations	
PK	LocationID
	LocationName
	SecondaryName

Battles	
PK	BattleID
FK	LocationID
	BattleYear
	BattleName
	BattleWon

Tables

1.Beings: The beings table includes common attributes that are shared amongst the subtypes Dragons, Orcs, Trolls, Elves, Goblins, Humans, Wizards, Hobbits, and Dwarves. (LocationID identifies the beings origin location).

```
--Beings—
create table Beings (
     CharacterID    char(4) not null,
     LocationID     char(4) not null references Locations(LocationID),
     TypeID         char(3) not null references TypeOfBeings(TypeID),
     Name           text,
     Title          text,
     Gender         char(6) not null check(gender='Male' or gender='Female'),
     BirthYear      char(7),
     DeathYear      char(6),
 primary key(CharacterID)
);
```



	characterid character(4)	locationid character(4)	typeid character(3)	name text	title text	gender character(6)	birthyear character(7)	deathyear character(6)
1	c001	L01	t08	Azog	The Defiler	Male	Unknown	TA2799
2	c002	L02	t01	Balin	Guardian of Erebor	Male	TA2763	TA2994
3	c003	L03	t04	Bard	The Bowman	Male	TA2898	TA2977
4	c004	L04	t10	Beorn	The SkinChanger	Male	Unknown	
5	c005	L05	t07	Bert	The Cook	Male	Unknown	TA2941

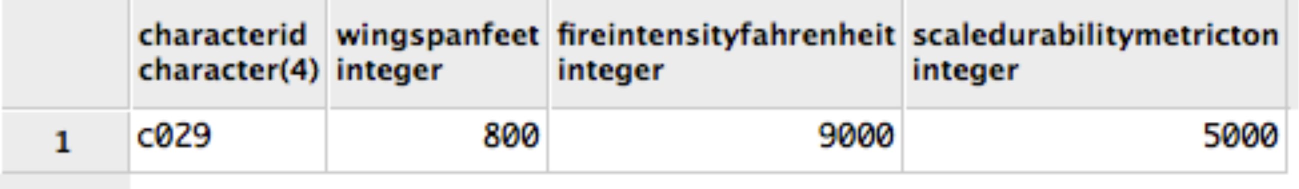
Functional Dependencies:

CharacterID → LocationID, TypeID, Name, Title, Gender, BirthYear, DeathYear

Tables

2.Dragons: Subtype of Beings. Contains all dragons and information that is specific to dragons such as their wing span, fire intensity, and scale durability. The measurements used are feet, Fahrenheit, and metric tons.

```
--Dragons—
Create table Dragons (
     CharacterID                    char(4) not null references Beings(CharacterID),
     WingSpanFeet                   Integer,
     FireIntensityFahrenheit        Integer,
     ScaleDurabilityMetricTon       integer,
 Primary key(CharacterID)
);
```



	characterid character(4)	wingspanfeet integer	fireintensityfahrenheit integer	scaledurabilitymetricton integer
1	c029	800	9000	5000

Functional Dependencies:

CharacterID → WingSpanFeet, FireIntensityFahrenheit, ScaleDurabilityMetricTon

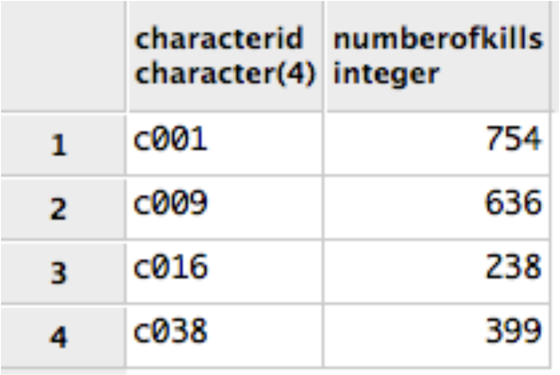
Tables

3.Orcs: Subtype of Beings. Contains all prevalent orcs and the amount of kills that they are notorious for.

```
--Orcs--
Create table Orcs (
     CharacterID          char(4) not null references Beings(CharacterID),
     NumberOfKills        integer,
 primary key(CharacterID)
);
```

Functional Dependencies:

CharacterID → NumberOfKills



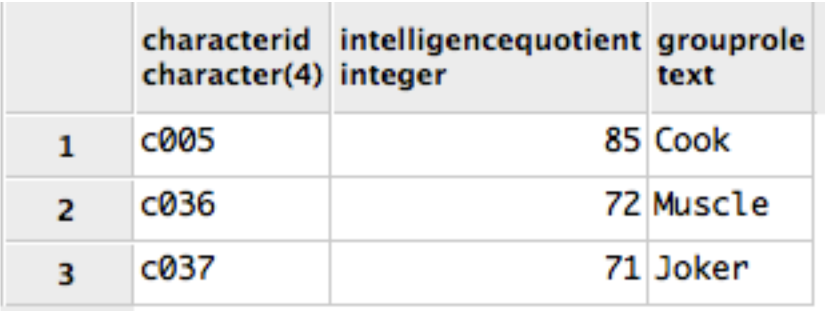
	characterid character(4)	numberofkills integer
1	c001	754
2	c009	636
3	c016	238
4	c038	399

4.Trolls: Subtype of Beings. Contains all trolls and troll related attributes such as their Intelligence Quotient (as a joke because they are known to be dim witted) and group role.

```
--Trolls—
Create table Trolls (
     CharacterID                char(4) not null references Beings(CharacterID),
     IntelligenceQuotient       Integer,
     GroupRole                  text,
 primary key(CharacterID)
);
```

Functional Dependencies:

CharacterID → IntelligenceQuotient, GroupRole

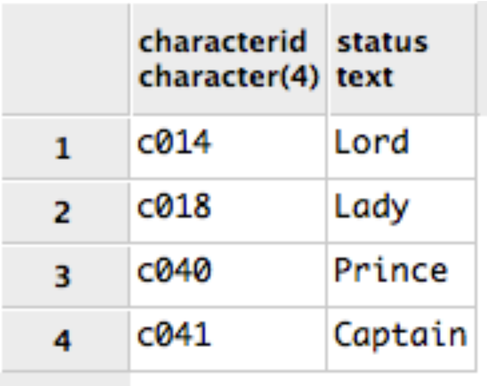


	characterid character(4)	intelligencequotient integer	grouprole text
1	c005	85	Cook
2	c036	72	Muscle
3	c037	71	Joker

Tables

5.Elves: Subtype of Beings. Contains all prevalent elves and their status (i.e. King, Prince, etc.)

```
--Elves—
Create table Elves (
     CharacterID          char(4) not null references Beings(CharacterID),
     Status               Text,
 primary key(CharacterID)
);
```



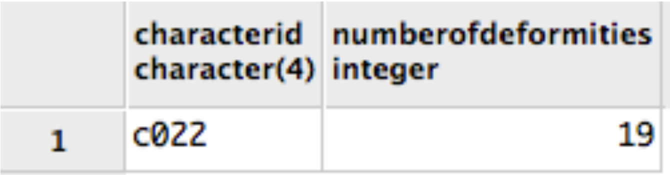
	characterid character(4)	status text
1	c014	Lord
2	c018	Lady
3	c040	Prince
4	c041	Captain

Functional Dependencies:

CharacterID → Status

6.Goblins: Subtype of Beings. Contains all prevalent goblins including the number of their deformities, as that is what they are known for.

```
--Goblins—
Create table Goblins (
CharacterID               char(4) not null references Beings(CharacterID),
NumberOfDeformities       Integer,
Primary key(CharacterID)
);
```



	characterid character(4)	numberofdeformities integer
1	c022	19

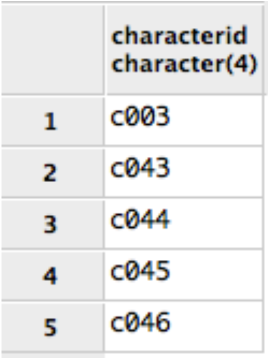
Functional Dependencies:

CharacterID → NumberOfDeformities

Tables

7.Humans: Subtype of Beings. In this case, humans are not especially none for anything so this table consists of only their associated CharacterID.

```
--Humans—
create table Humans (
     CharacterID          char(4) not null references Beings(CharacterID),
 primary key(CharacterID)
);
```



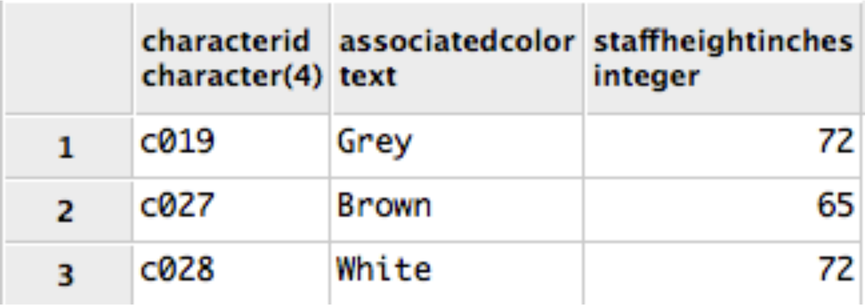
	characterid character(4)
1	c003
2	c043
3	c044
4	c045
5	c046

Functional Dependencies:

CharacterID → CharacterID

8.Wizards: Subtype of Beings. Contains all prevalent wizards and their associated color and staff heights (measured in inches).

```
--Wizards—
Create table Wizards (
     CharacterID          char(4) not null references Beings(CharacterID),
     AssociatedColor      text,
     StaffHeightInches    integer,
 primary key(CharacterID)
);
```



	characterid character(4)	associatedcolor text	staffheightinches integer
1	c019	Grey	72
2	c027	Brown	65
3	c028	White	72

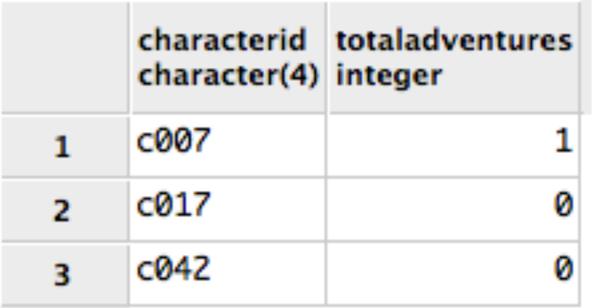
Functional Dependencies:

CharacterID → AssociatedColor, StaffHeightInches

Tables

9.Hobbits: Subtype of Beings. Contains all prevalent hobbits along with their total adventures to point out its rarity.

```
--Hobbits—
Create table Hobbits (
     CharacterID          char(4) not null references Beings(CharacterID),
     TotalAdventures      integer,
 primary key(CharacterID)
);
```



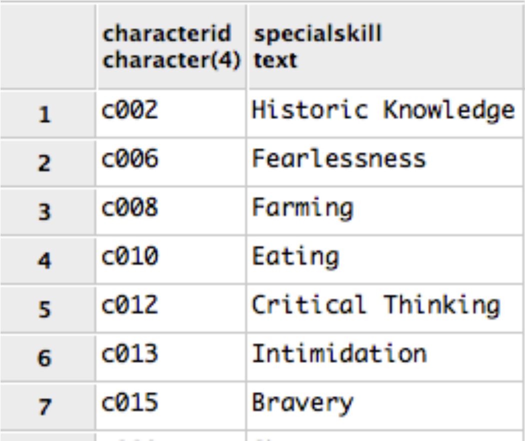
	characterid character(4)	totaladventures integer
1	c007	1
2	c017	0
3	c042	0

Functional Dependencies:

CharacterID → TotalAdventures

10.Dwarves: Subtype of Beings. Contains all prevalent dwarves and their special skill set that they offer.

```
--Dwarves—
Create table Dwarves (
     CharacterID          char(4) not null references Beings(CharacterID),
     SpecialSkill         text,
 primary key(CharacterID)
);
```



	characterid character(4)	specialskill text
1	c002	Historic Knowledge
2	c006	Fearlessness
3	c008	Farming
4	c010	Eating
5	c012	Critical Thinking
6	c013	Intimidation
7	c015	Bravery

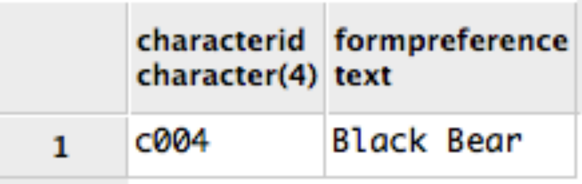
Functional Dependencies:

CharacterID → SpecialSkill

Tables

11.SkinChangers: Subtype of Beings. Contains all mentioned and encountered Skinchangers and the form the prefer to be in.

```
--SkinChangers—
create table SkinChangers (
     CharacterID          char(4) not null references Beings(CharacterID),
     FormPreference       text,
 primary key(CharacterID)
);
```



	characterid character(4)	formpreference text
1	c004	Black Bear

Functional Dependencies:

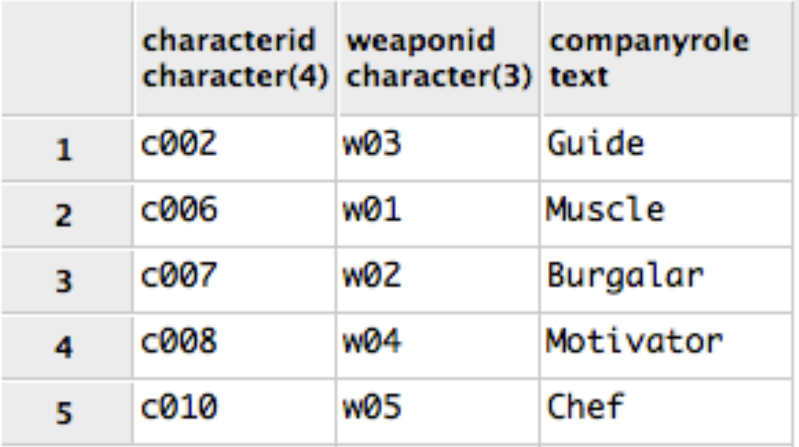
CharacterID → FormPreference

12.CompanyMembers: The company consists of a wizard, a hobbits, and dwarves. This table also includes each members weapon of choice (WeaponID) and their role in The Company.

```
--CompanyMembers—
Create table CompanyMembers (
     CharacterID          char(4) not null references Beings(CharacterID),
     WeaponID             char(3),
     CompanyRole          text,
 primary key(CharacterID)
);
```

Functional Dependencies:

CharacterID→ WeaponID, CompanyRole

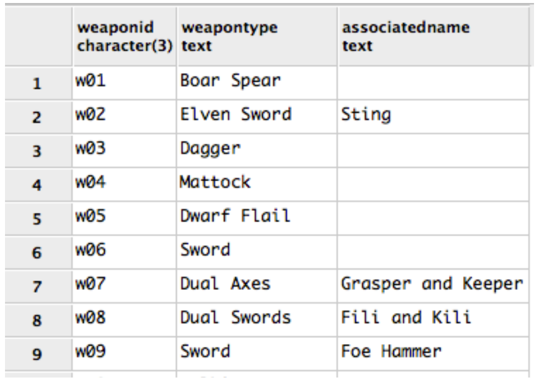


	characterid character(4)	weaponid character(3)	companyrole text
1	c002	w03	Guide
2	c006	w01	Muscle
3	c007	w02	Burgalar
4	c008	w04	Motivator
5	c010	w05	Chef

Tables

<u>13.Weapons:</u> The weapons table consists of all of the weapons of choice for the company members and any associated name with the weapon.

```
--Weapons—
create table Weapons (
     WeaponID       char(3) not null,
     TypeOfWeapon         text,
     AssociatedName       text,
 primary key(WeaponID)
);
```



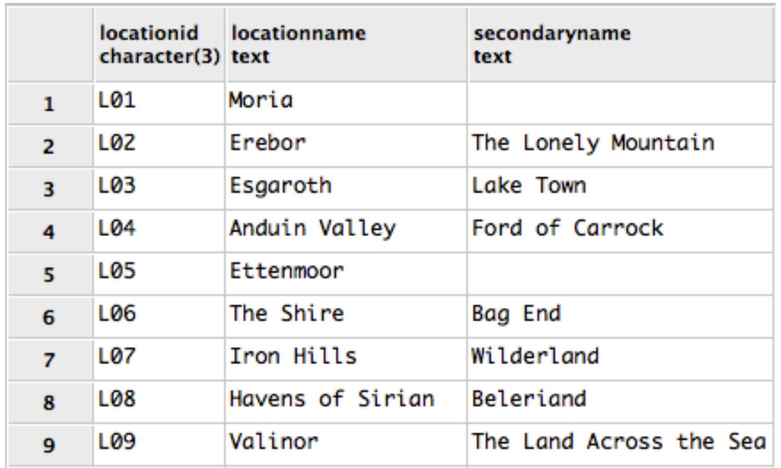
	weaponid character(3)	weapontype text	associatedname text
1	w01	Boar Spear	
2	w02	Elven Sword	Sting
3	w03	Dagger	
4	w04	Mattock	
5	w05	Dwarf Flail	
6	w06	Sword	
7	w07	Dual Axes	Grasper and Keeper
8	w08	Dual Swords	Fili and Kili
9	w09	Sword	Foe Hammer

Functional Dependencies:

WeaponID→ TypeOfWeapon, AssociatedName

<u>14.Locations:</u> Contains all locations relevant to being's origin locations and battle locations.

```
--Locations—
create table Locations (
     LocationID          char(3) not null,
     LocationName        text,
     SecondaryName       text,
 primary key(LocationID)
);
```



	locationid character(3)	locationname text	secondaryname text
1	L01	Moria	
2	L02	Erebor	The Lonely Mountain
3	L03	Esgaroth	Lake Town
4	L04	Anduin Valley	Ford of Carrock
5	L05	Ettenmoor	
6	L06	The Shire	Bag End
7	L07	Iron Hills	Wilderland
8	L08	Havens of Sirian	Beleriand
9	L09	Valinor	The Land Across the Sea

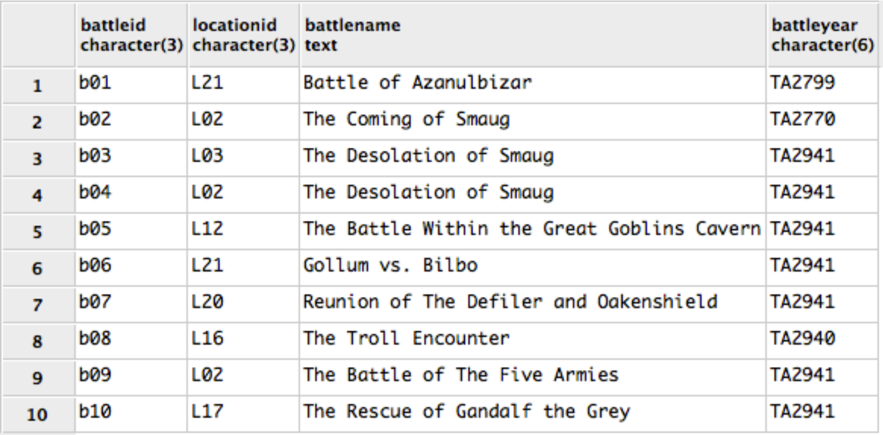
Functional Dependencies:

LocationID → LocationName, SecondaryName

Tables

15.Battles: Contains all battles that were experienced, the common factor being the company members.

```
--Battles--
create table Battles (
     BattleID       char(3) not null,
     LocationID          char(3) not null references Locations(LocationID),
     BattleName          text,
     BattleYear          integer,
     BattleWon      boolean,
 primary key(BattleID)
);
```



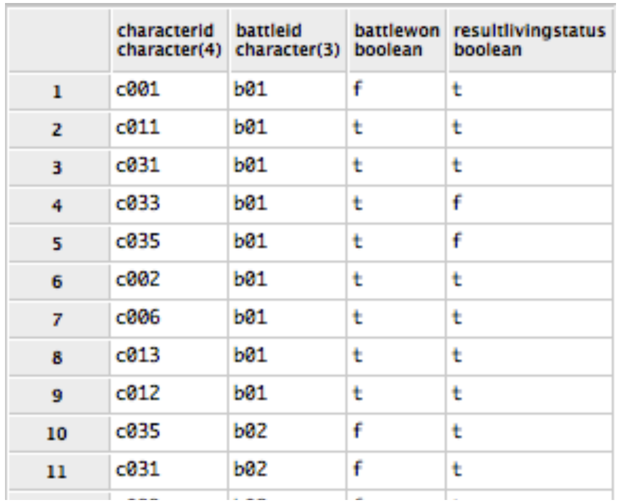
	battleid character(3)	locationid character(3)	battlename text	battleyear character(6)
1	b01	L21	Battle of Azanulbizar	TA2799
2	b02	L02	The Coming of Smaug	TA2770
3	b03	L03	The Desolation of Smaug	TA2941
4	b04	L02	The Desolation of Smaug	TA2941
5	b05	L12	The Battle Within the Great Goblins Cavern	TA2941
6	b06	L21	Gollum vs. Bilbo	TA2941
7	b07	L20	Reunion of The Defiler and Oakenshield	TA2941
8	b08	L16	The Troll Encounter	TA2940
9	b09	L02	The Battle of The Five Armies	TA2941
10	b10	L17	The Rescue of Gandalf the Grey	TA2941

Functional Dependencies:

BattleID → LocationID, BattleName, BattleYear, BattleWon

16.FoughtIn: Contains the associated characters for each battle and whether they survived or not.

```
--FoughtIn—
create table FoughtIn (
     CharacterID         char(4) not null references Beings(CharacterID),
     BattleID            char(3) not null references Battles(BattleID),
     ResultLivingStatus  Boolean,
 primary key(CharacterID, BattleID)
);
```



	characterid character(4)	battleid character(3)	battlewon boolean	resultlivingstatus boolean
1	c001	b01	f	t
2	c011	b01	t	t
3	c031	b01	t	t
4	c033	b01	t	f
5	c035	b01	t	f
6	c002	b01	t	t
7	c006	b01	t	t
8	c013	b01	t	t
9	c012	b01	t	t
10	c035	b02	f	t
11	c031	b02	f	t

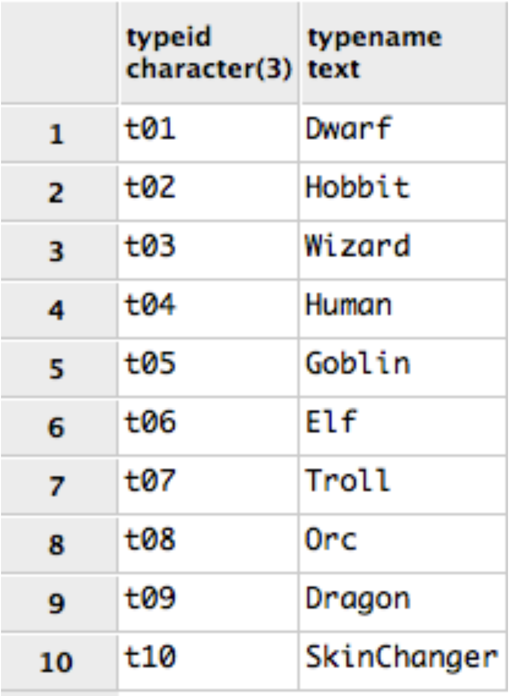
Functional Dependencies:

CharacterID, BattleID → ResultLivingStatus

Tables

16.TypesOfBeings: Contains all battles that were experienced, the common factor being the company members.

```
--TypeOfBeings—
create table TypeOfBeings (
     TypeID    char(3) not null,
     TypeName  text not null,
 primary key(TypeID)
);
```



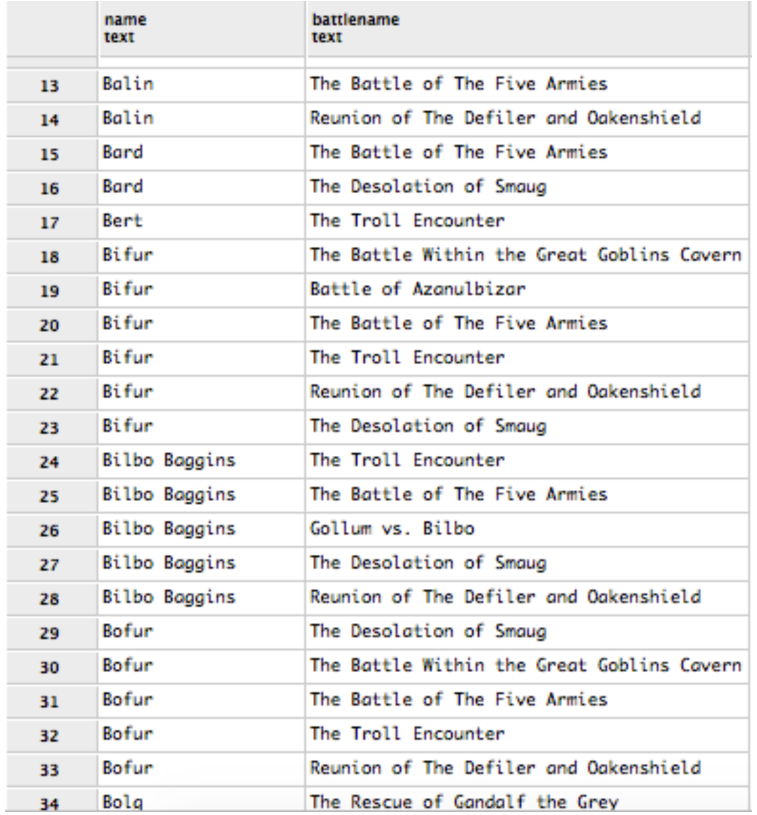
	typeid character(3)	typename text
1	t01	Dwarf
2	t02	Hobbit
3	t03	Wizard
4	t04	Human
5	t05	Goblin
6	t06	Elf
7	t07	Troll
8	t08	Orc
9	t09	Dragon
10	t10	SkinChanger

Functional Dependencies:

TypeID → TypeName

Views

1. **BattleRecord:** Lists the names of characters and what battles they participated in.



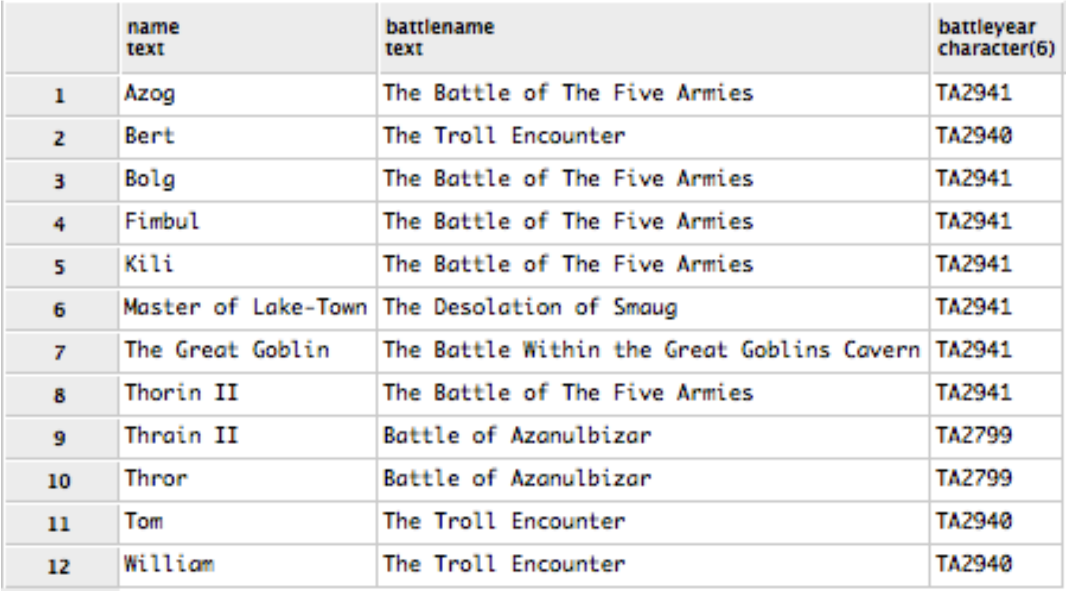
	name text	battlename text
13	Balin	The Battle of The Five Armies
14	Balin	Reunion of The Defiler and Oakenshield
15	Bard	The Battle of The Five Armies
16	Bard	The Desolation of Smaug
17	Bert	The Troll Encounter
18	Bifur	The Battle Within the Great Goblins Cavern
19	Bifur	Battle of Azanulbizar
20	Bifur	The Battle of The Five Armies
21	Bifur	The Troll Encounter
22	Bifur	Reunion of The Defiler and Oakenshield
23	Bifur	The Desolation of Smaug
24	Bilbo Baggins	The Troll Encounter
25	Bilbo Baggins	The Battle of The Five Armies
26	Bilbo Baggins	Gollum vs. Bilbo
27	Bilbo Baggins	The Desolation of Smaug
28	Bilbo Baggins	Reunion of The Defiler and Oakenshield
29	Bofur	The Desolation of Smaug
30	Bofur	The Battle Within the Great Goblins Cavern
31	Bofur	The Battle of The Five Armies
32	Bofur	The Troll Encounter
33	Bofur	Reunion of The Defiler and Oakenshield
34	Bolg	The Rescue of Gandalf the Grey

```
--BattleRecord—
create view BattleRecord as
  select b.Name as Name,
  ba.BattleName
  from FoughtIn f inner join Beings b on f.CharacterID  = b.CharacterID
                  inner join Battles ba on f.BattleID = ba.BattleID
order by name ASC;
```

Views

2. KilledInAction: Lists the name of characters, the name of the battle, and the year in which the battle took place of when the character was killed in action.

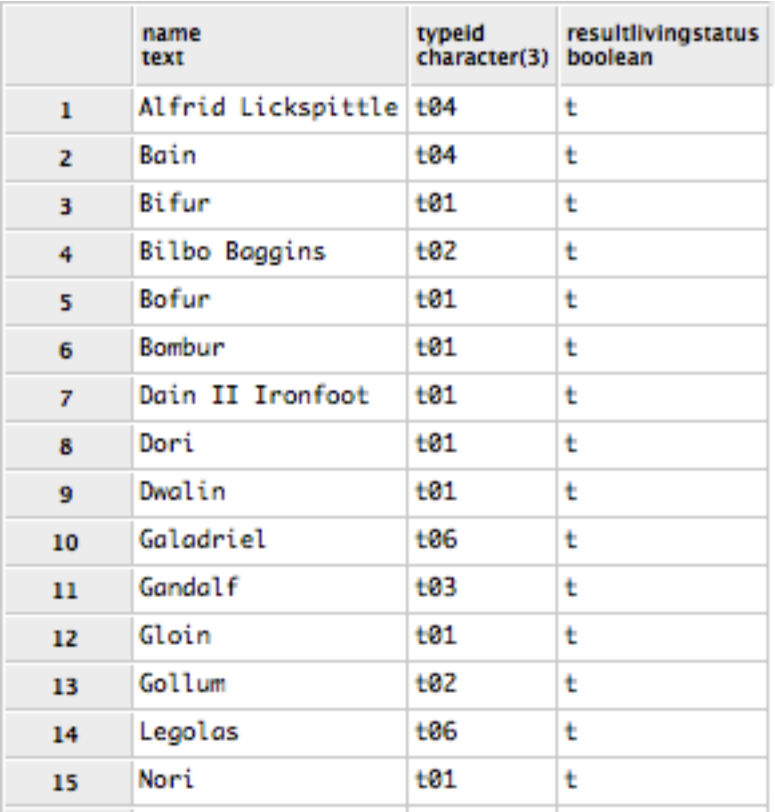
```
--KilledInAction—
create view KilledInAction as
    select b.Name,
           ba.BattleName,
           ba.BattleYear
    from FoughtIn f inner join Beings b on f.CharacterID = b.CharacterID
                    inner join Battles ba on f.BattleID = ba.BattleID
    where f.ResultLivingStatus = 'f’
order by name asc;
```



	name text	battlename text	battleyear character(6)
1	Azog	The Battle of The Five Armies	TA2941
2	Bert	The Troll Encounter	TA2940
3	Bolg	The Battle of The Five Armies	TA2941
4	Fimbul	The Battle of The Five Armies	TA2941
5	Kili	The Battle of The Five Armies	TA2941
6	Master of Lake-Town	The Desolation of Smaug	TA2941
7	The Great Goblin	The Battle Within the Great Goblins Cavern	TA2941
8	Thorin II	The Battle of The Five Armies	TA2941
9	Thrain II	Battle of Azanulbizar	TA2799
10	Thror	Battle of Azanulbizar	TA2799
11	Tom	The Troll Encounter	TA2940
12	William	The Troll Encounter	TA2940

Views

3. LivingStatus: Lists all the living characters that have neither died from natural causes of battle.



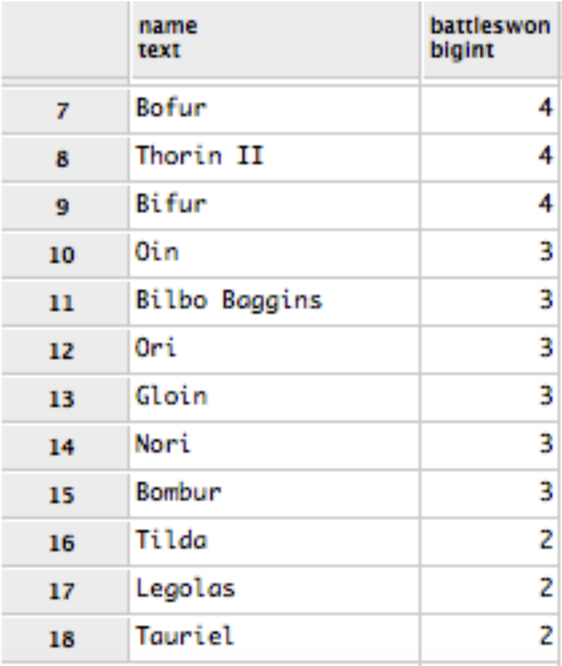
	name text	typeid character(3)	resultlivingstatus boolean
1	Alfrid Lickspittle	t04	t
2	Bain	t04	t
3	Bifur	t01	t
4	Bilbo Baggins	t02	t
5	Bofur	t01	t
6	Bombur	t01	t
7	Dain II Ironfoot	t01	t
8	Dori	t01	t
9	Dwalin	t01	t
10	Galadriel	t06	t
11	Gandolf	t03	t
12	Gloin	t01	t
13	Gollum	t02	t
14	Legolas	t06	t
15	Nori	t01	t

```
--LivingStatus—
view LivingStatus as
    select distinct   b.Name,
                     b.TypeID,
                     f.ResultLivingStatus
    from FoughtIn f inner join Beings b on f.CharacterID = b.CharacterID
                      inner join Battles ba on f.BattleID = ba.BattleID
    where f.ResultLivingStatus = 't'
    and   b.DeathYear is null
order by name ASC;
```

Reports and Interesting Queries

1. Query to return the number of battles one by each person.

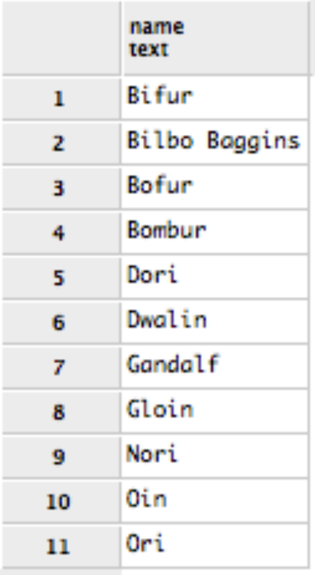
```
select b.Name,
count(b.CharacterID) as BattlesWon
from FoughtIn f inner join Beings b on f.CharacterID = b.CharacterID
inner join Battles ba on f.BattleID = ba.BattleID
where f.BattleWon = 't’
group by b.Name
order by BattlesWon DESC;
```



	name text	battleswon bigint
7	Bofur	4
8	Thorin II	4
9	Bifur	4
10	Oin	3
11	Bilbo Baggins	3
12	Ori	3
13	Gloin	3
14	Nori	3
15	Bombur	3
16	Tilda	2
17	Legolas	2
18	Tauriel	2

2. Query to return the age of each being.

```
select b.name
from Beings b inner join CompanyMembers cm on b.CharacterID = cm.CharacterID
where b.DeathYear is nullgroup by b.CharacterID
Order by b.name ASC;
```



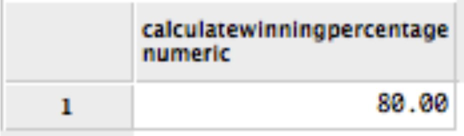
	name text
1	Bifur
2	Bilbo Baggins
3	Bofur
4	Bombur
5	Dori
6	Dwalin
7	Gandalf
8	Gloin
9	Nori
10	Oin
11	Ori

Stored Procedures

CalulateWinningPercentage: For the stated Being’s name, the stored procedure calculates the percentage of battles won.

```
--CalculateWinningPercentage
create or replace function CalculateWinningPercentage(CombatantName text)
returns table(WinningPercentage numeric) as
$$
begin
Return query select trunc (
(cast(
(select count(f.BattleWon)
from FoughtIn f inner join Beings b on f.CharacterID = b.CharacterID
inner join Battles ba on f.BattleID = ba.BattleID
where b.Name = CombatantName
and   f.BattleWon = 't'
) as decimal (5,2)
)
/
(select count(f.BattleWon) as TotalBattlesFought
from FoughtIn f inner join Beings b on f.CharacterID = b.CharacterID
inner join Battles ba on f.BattleID = ba.BattleID
where b.Name = CombatantName
)*100), 2
)as WinningPercentage
From Beings
where name = CombatantName;end;
$$
language plpgsql;
```

```
select CalculateWinningPercentage('Thorin II');
```



	calculatewinningpercentage numeric
1	80.00

Stored Procedure

DeathUpdate: The trigger sets the DeathYear of a being to the current year if the ResultLivingStatus is changed due to the result of a Battle.

```
create or replace function DeathUpdate() returns trigger as
$$
declare CurrentYear string := 'TA2942';
begin
if new.ResultLivingStatus = false
and (select DeathYear
from Beings
where ResultLivingStatus = new.ResultLivingStatus) is null
then update Beings
 set DeathYear = CurrentYear
where ResultLivingStatus = new.ResultLivingStatus
end if;
return new;
end;
$$
language plpgsql;
```

Trigger

```
 --DeathUpdate
 create trigger DeathUpdate
 after update on FoughtIn
 for each row execute DeathUpdate();
```

Security

Administrator Role: The role of the database administrator with full access to the database.

```
create role admin;
grant all on all tables in schema public to admin;
```

Baggins Role: The role of Bilbo Baggins and Frodo Baggins; able to access and change data to keep information up to date, as Frodo is expected to add his own adventures to the database.

```
Create role Baggins;
grant select on all tables in schema public to admin;
grant insert on Beings, TypeOfBeings, SkinChangers, Dragons,
Orcs, Trolls, Elves, Goblins, Humans, Wizards,
Hobbits, Dwarves, FoughtIn, Battles, Locations to Baggins;
grant update on Beings, TypeOfBeings, SkinChangers, Dragons,
Orcs, Trolls, Elves, Goblins, Humans, Wizards,
Hobbits, Dwarves, FoughtIn, Battles, Locations,
CompanyMembers, Weapons to Baggins;
```

Implementation Notes

- The date structure that Middle Earth uses is not supported by PostGres. Middle Earth includes the present age (First, Second, Third) and restarts the counting of the years. A new date structure will have to be constructed to make all fields atomic
- Everything included in this database is based off of information that Bilbo Baggins has collected through stories and experience.
- Battle names do not have to be unique as they are often named after the location at which it took place.
- Predecessors of the Lonely Mountain are given the same first names as a sign of a confident lineage, so the names can be the same.
- Characters accounted for during battles do not necessarily possess fighting skills. For example, Alfrid Lipstickle has survived several battles by avoiding fighting, thus it cannot be concluded that the number of battles won cannot be correlated with the character’s fight abilities.

Known Problems/ Future Enhancements

- To determine the age of some the characters, it may present an issue as some birthdates are not known.
- The data collected for this database is purely based on the Baggins' knowledge, so the accuracy can be questionable
- Determining the type of each character through the TypeOfBeings table is fairly easy, but distributing that same information to the correlated subtype may present a challenge. The specific subtype demands specific traits to be noted
- In the future, more types of beings will be encountered, thus making the table more complex. Creating a specific subtype table for the type of being may present a problem because it calls for the creation new tables.
- The Baggins' are given almost identical access to the database as the admin. So if new information is found, such as the birthdate of Gandalf, his age will be able to be determined.

Thank you