A DATABASE
OF
ICE AND FIRE

DESIGNED BY LIAM HARWOOD
 Table of Contents:

Executive Summary .................................................................................................................. 3
Entity Relationship Diagram .................................................................................................. 4
Tables .................................................................................................................................... 5
Views ..................................................................................................................................... 14
Reports and Interesting Queries .............................................................................................. 18
Stored Procedures .................................................................................................................. 20
Triggers .................................................................................................................................. 23
Security .................................................................................................................................. 25
Implementation Notes ............................................................................................................ 27
Known Problems / Future Enhancements ............................................................................... 28
Executive Summary

This document outlines the design and implementation of a database which holds up-to-date historical data on the people, houses, and conflicts of Westeros. First, the ER diagram of the database will be presented, followed by a description of each table and its SQL create statements. Sample data for each table will also be shown. Next, the views, reports, stored procedures, and triggers implemented in this database will be discussed, along with sample output. Then, there will be a description of all roles and security privileges suggested for this database. Finally, notes on the implementation will be given, followed by known issues and ideas for future improvement.

This implementation is intended to be used by the maesters of the Citadel for the purpose of furthering the documentation and analysis of Westerosi history. This database will allow for easier keeping of records concerning the people, places, and events of Westeros. Maesters will be able to gain useful information from queries that will provide interesting and valuable historical data. The goal of this database implementation is to provide the Citadel with a functional, normalized database that will make the process of keeping track of historical events (past, present, and future) easier and more efficient.
**TABLES**

**People:** The people table contains all people and their basic attributes, which are shared with the subtypes Kings, Knights, and Night’s Watchmen.

```sql
CREATE TABLE People (  
  pid integer NOT NULL,  
  name text NOT NULL,  
  gender char(1) NOT NULL check(gender='m' or gender='f'),  
  birthyear integer NOT NULL,  
  deathyear integer,  
  primary key(pid)  
);
```

Functional Dependencies:

\[ pid \rightarrow name, gender, birthyear, deathyear \]

Sample Data 1: People

```
<table>
<thead>
<tr>
<th>pid</th>
<th>integer</th>
<th>name</th>
<th>gender</th>
<th>birthyear</th>
<th>deathyear</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Eddard Stark</td>
<td>m</td>
<td>263</td>
<td>299</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Robb Stark</td>
<td>m</td>
<td>263</td>
<td>299</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Catelyn Stark</td>
<td>f</td>
<td>264</td>
<td>299</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Sansa Stark</td>
<td>f</td>
<td>286</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Arya Stark</td>
<td>f</td>
<td>289</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Euron Greyjoy</td>
<td>m</td>
<td>267</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Tyrion Lannister</td>
<td>m</td>
<td>273</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Tywin Lannister</td>
<td>m</td>
<td>242</td>
<td>300</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Jaime Lannister</td>
<td>m</td>
<td>266</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Cersei Lannister</td>
<td>f</td>
<td>266</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Daenerys Targaryen</td>
<td>f</td>
<td>234</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Robert Baratheon</td>
<td>m</td>
<td>262</td>
<td>298</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Stannis Baratheon</td>
<td>m</td>
<td>264</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Joffrey Baratheon</td>
<td>m</td>
<td>236</td>
<td>300</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Jon Snow</td>
<td>m</td>
<td>283</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Davos Seaworth</td>
<td>m</td>
<td>260</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Brienne of Tarth</td>
<td>f</td>
<td>230</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Samwell Tarly</td>
<td>m</td>
<td>233</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Oberyn Martell</td>
<td>m</td>
<td>257</td>
<td>300</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Doran Martell</td>
<td>m</td>
<td>247</td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Theon Greyjoy</td>
<td>m</td>
<td>278</td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>Balon Greyjoy</td>
<td>m</td>
<td>250</td>
<td>299</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>Tommen Baratheon</td>
<td>m</td>
<td>291</td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>Edmure Tully</td>
<td>m</td>
<td>270</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```
**Kings:** Subtype of People. Contains all people that are kings and basic information about their reign.

```sql
create table Kings (  
    pid integer not null references People(pid),  
    title text not null,  
    reignstartyear integer not null,  
    reignendyear integer,  
    primary key(pid)
);
```

Functional Dependencies:

- `pid` → `title`, `reignstartyear`, `reignendyear`

**Knights:** Subtype of People. Contains all people that are knights and the type of knight that each person is.

```sql
create table Knights (  
    pid integer not null references People(pid),  
    type text,  
    primary key(pid)
);
```

Functional Dependencies:

- `pid` → `type`
**Night’s Watchmen:** Subtype of People. Contains all people that are members of the Night’s Watch as well as their job in the Watch.

```sql
create table NightsWatchMen (  
    pid integer not null references People(pid),  
    job text    check(job='ranger' or job='steward' or job='builder'),  
    primary key(pid)
);
```

**Functional Dependencies:**

\[ \text{pid} \rightarrow \text{job} \]

**Aliases:** Contains any aliases, nicknames, or assumed identities that people have used.

```sql
create table Aliases (  
    pid integer not null references People(pid),  
    name text     not null,  
    primary key(pid, name)
);
```

**Functional Dependencies:**

\[ (\text{pid, name}) \rightarrow \]
**Regions:** Contains the different geographic regions of Westeros.

```sql
create table Regions (  
  rid integer not null,  
  name text    not null,  
  description text    not null,  
  primary key(rid)  
);
```

Functional Dependencies:

rid $\rightarrow$ name, description

**Castles:** Contains the different castles and strongholds as well as their locations.

```sql
create table Castles (  
  cid integer not null,  
  name text    not null,  
  rid integer not null references Regions(rid),  
  primary key(cid)  
);
```

Functional Dependencies:

cid $\rightarrow$ name, rid
Houses: Contains the Great Houses (families) of Westeros as well as basic information about each house. (Note: “seatid” is a foreign key referencing the ID of the current castle where the house is based and “lordid” is a foreign key referencing the ID of the person who is currently the lord of the house.)

cREATE TABLE Houses (  
  hid integer not null,  
  name text not null,  
  words text,  
  coatofarms text,  
  seatid integer references Castles(cid),  
  lordid integer references People(pid),  
  primary key(hid)  
);

Functional Dependencies:

Hid → name, words, coatofarms, seatid, lordid

<table>
<thead>
<tr>
<th>hid</th>
<th>name</th>
<th>words</th>
<th>coatofarms</th>
<th>seatid</th>
<th>lordid</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Stark</td>
<td>Winter is Coming</td>
<td>Direwolf</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Lannister</td>
<td>Hear Me Roar</td>
<td>Lion</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>Baratheon of Dragonstone</td>
<td>Ours is the Fury</td>
<td>Flaming Stag</td>
<td>3</td>
<td>13</td>
</tr>
<tr>
<td>4</td>
<td>Greyjoy</td>
<td>We Do Not Sow</td>
<td>Kraken</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>Martell</td>
<td>Unbowed, Unbent, Unbroken</td>
<td>Sun and spear</td>
<td>5</td>
<td>20</td>
</tr>
<tr>
<td>6</td>
<td>Tully</td>
<td>Family, Duty, Honor</td>
<td>Trout</td>
<td>6</td>
<td>24</td>
</tr>
<tr>
<td>7</td>
<td>Targaryen</td>
<td>Fire and Blood</td>
<td>Dragon</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Baratheon of King's Landing</td>
<td>Ours is the Fury</td>
<td>Stag and lion</td>
<td>10</td>
<td>23</td>
</tr>
</tbody>
</table>

Sample Data 8: Houses
Allegiance: Contains the current allegiances of each person, i.e. the houses that they are loyal to, as well as the nature of this allegiance (familial, marital, or sworn vow).

create table Allegiance (
    pid  integer not null references People(pid),
    hid  integer not null references Houses(hid),
    type text    check(type='marital' or type='familial' or type='sworn'),
    primary key(pid, hid)
);

Functional Dependencies:
(pid, hid) → type

Wars: Contains different wars that have happened or are currently happening, as well as some basic information about each war.

create table Wars (
    wid        integer not null,
    name       text    not null,
    startyear  integer not null,
    endyear    integer,
    primary key(wid)
);

Functional Dependencies:
wid → name, startyear, endyear
**Factions:** Contains the factions that were in conflict in each war as well as basic information about each faction. (Note: “wonwar” is true if the faction won the war, false if the faction lost the war, and null if the war is ongoing.)

```sql
CREATE TABLE Factions(
    fid integer NOT NULL,
    wid integer NOT NULL REFERENCES Wars(wid),
    name text NOT NULL,
    wonwar boolean,
    PRIMARY KEY(fid)
);
```

Functional Dependencies:

fid \(\rightarrow\) wid, name, wonwar

**AlliedWith:** Contains the factions that each house was allied with in the different wars.

```sql
CREATE TABLE AlliedWith(
    hid integer NOT NULL REFERENCES Houses(hid),
    fid integer NOT NULL REFERENCES Factions(fid),
    PRIMARY KEY(hid, fid)
);
```

Functional Dependencies:

(hid, fid) \(\rightarrow\)
**Battles:** Contains the battles that were fought during the different wars as well as basic information about each battle.

```sql
create table Battles (  
    bid integer not null,  
    wid integer not null references Wars(wid),  
    rid integer not null references Regions(rid),  
    name text not null,  
    primary key(bid)
);  
```

Functional Dependencies:

`bid → wid, rid, name`

**Combatants:** Contains the factions that fought in each battle as well as whether or not they won the battle.

```sql
create table Combatants (  
    bid integer not null references Battles(bid),  
    fid integer not null references Factions(fid),  
    wonbattle boolean,  
    primary key(bid, fid)
);  
```

Functional Dependencies:

`(bid, fid) → wonbattle`
**FoughtIn:** Contains the people that fought in each battle as well as the faction they fought for and whether or not they died at that battle.

```
cREATE TABLE FoughtIn (  
  bid  integer not null references Battles(bid),  
  pid  integer not null references People(pid),  
  fid  integer not null references Factions(fid),  
  died boolean not null,  
  primary key(pid, bid)  
);```

**Functional Dependencies:**

\((pid, bid) \rightarrow fid, died\)
**Views**

**LivingKings:** Lists the names and titles of all living, reigning kings

create view LivingKings as

    select name, title
    from people p inner join kings k on p.pid = k.pid
    where deathyear is null
        and reignendyear is null
    order by name asc;

<table>
<thead>
<tr>
<th>Name</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>Euron Greyjoy</td>
<td>Iron King of the Isles and the North</td>
</tr>
<tr>
<td>Tommen Baratheon</td>
<td>King of the Andals, the Rhoynar and the First Men</td>
</tr>
</tbody>
</table>

*Sample Output 1: LivingKings*
**BattleExperience**: Lists the names of all people along with the wars they fought in, each battle they fought in for each war, and the side they were on for each battle.

```
create view BattleExperience as
    select p.name as person, w.name as war, b.name as battle, fa.name as faction
    from   people p inner join foughtin fi on p.pid  = fi.pid
    inner join factions fa on fi.fid = fa.fid
    inner join battles  b  on fi.bid = b.bid
    inner join wars     w  on b.wid  = w.wid
    order by person asc;
```

### Sample Output 2: BattleExperience

<table>
<thead>
<tr>
<th>person</th>
<th>war text</th>
<th>battle text</th>
<th>faction text</th>
</tr>
</thead>
<tbody>
<tr>
<td>Catelyn Stark</td>
<td>War of the Five Kings</td>
<td>The Red Wedding</td>
<td>The King in the North and the Trident</td>
</tr>
<tr>
<td>Daenerys Targaryen</td>
<td>War of the Five Kings</td>
<td>Siege of Storm's End</td>
<td>The King in the Narrow Sea</td>
</tr>
<tr>
<td>Daenerys Targaryen</td>
<td>War of the Five Kings</td>
<td>Battle of the Blackwater</td>
<td>The King in the Narrow Sea</td>
</tr>
<tr>
<td>Davos Seaworth</td>
<td>Conflict Beyond the Wall</td>
<td>Battle of Castle Black</td>
<td>The Night's Watch</td>
</tr>
<tr>
<td>Eddard Stark</td>
<td>Robert's Rebellion</td>
<td>Tower of Joy</td>
<td>Rebels</td>
</tr>
<tr>
<td>Eddard Stark</td>
<td>Robert's Rebellion</td>
<td>Sack of King's Landing</td>
<td>Rebels</td>
</tr>
<tr>
<td>Eddard Stark</td>
<td>Robert's Rebellion</td>
<td>Battle of the Trident</td>
<td>Rebels</td>
</tr>
<tr>
<td>Euron Greyjoy</td>
<td>War of the Five Kings</td>
<td>Battle near River Run</td>
<td>The King in the North and the Trident</td>
</tr>
<tr>
<td>Euron Greyjoy</td>
<td>War of the Five Kings</td>
<td>The Red Wedding</td>
<td>The King in the North and the Trident</td>
</tr>
<tr>
<td>Jaime Lannister</td>
<td>War of the Five Kings</td>
<td>Battle near the Golden Tooth</td>
<td>The King in the Iron Throne</td>
</tr>
<tr>
<td>Jaime Lannister</td>
<td>War of the Five Kings</td>
<td>Battle near River Run</td>
<td>The King in the Iron Throne</td>
</tr>
<tr>
<td>Jaime Lannister</td>
<td>War of the Five Kings</td>
<td>Battle in the Whispering Wood</td>
<td>The King in the Iron Throne</td>
</tr>
<tr>
<td>Jaime Lannister</td>
<td>Sack of King's Landing</td>
<td>Rebels</td>
<td></td>
</tr>
<tr>
<td>Jon Snow</td>
<td>Conflict Beyond the Wall</td>
<td>Battle of Castle Black</td>
<td>The Night's Watch</td>
</tr>
<tr>
<td>Jon Snow</td>
<td>Conflict Beyond the Wall</td>
<td>Fight at the Flea</td>
<td>The Night's Watch</td>
</tr>
<tr>
<td>Robb Stark</td>
<td>War of the Five Kings</td>
<td>The Red Wedding</td>
<td>The King in the North and the Trident</td>
</tr>
<tr>
<td>Robb Stark</td>
<td>War of the Five Kings</td>
<td>Battle in the Whispering Wood</td>
<td>The King in the North and the Trident</td>
</tr>
<tr>
<td>Robb Stark</td>
<td>War of the Five Kings</td>
<td>Battle of Osasco</td>
<td>The King in the North and the Trident</td>
</tr>
<tr>
<td>Robb Stark</td>
<td>War of the Five Kings</td>
<td>Battle of the Camps</td>
<td>The King in the North and the Trident</td>
</tr>
<tr>
<td>Robert Baratheon</td>
<td>Robert's Rebellion</td>
<td>Battle of the Trident</td>
<td>Rebels</td>
</tr>
<tr>
<td>Samwell Tarly</td>
<td>Conflict Beyond the Wall</td>
<td>Battle of Castle Black</td>
<td>The Night's Watch</td>
</tr>
<tr>
<td>Samwell Tarly</td>
<td>Conflict Beyond the Wall</td>
<td>Fight at the Flea</td>
<td>The Night's Watch</td>
</tr>
<tr>
<td>Stannis Baratheon</td>
<td>Conflict Beyond the Wall</td>
<td>Battle of Castle Black</td>
<td>The Night's Watch</td>
</tr>
<tr>
<td>Stannis Baratheon</td>
<td>War of the Five Kings</td>
<td>Siege of Storm's End</td>
<td>The King in the Narrow Sea</td>
</tr>
<tr>
<td>Stannis Baratheon</td>
<td>War of the Five Kings</td>
<td>Battle of the Blackwater</td>
<td>The King in the Narrow Sea</td>
</tr>
<tr>
<td>Theon Greyjoy</td>
<td>War of the Five Kings</td>
<td>Battle at Winterfell</td>
<td>The King of the Isles and the North</td>
</tr>
<tr>
<td>Tyrion Lannister</td>
<td>War of the Five Kings</td>
<td>Battle of the Blackwater</td>
<td>The King on the Iron Throne</td>
</tr>
<tr>
<td>Tywin Lannister</td>
<td>War of the Five Kings</td>
<td>Battle on the Green Fork</td>
<td>The King on the Iron Throne</td>
</tr>
<tr>
<td>Tywin Lannister</td>
<td>Robert's Rebellion</td>
<td>Sack of King's Landing</td>
<td>Rebels</td>
</tr>
<tr>
<td>Tywin Lannister</td>
<td>War of the Five Kings</td>
<td>Battle of the Blackwater</td>
<td>The King on the Iron Throne</td>
</tr>
</tbody>
</table>
HouseWarHistory: Lists the names of all houses along with the wars they were a part of and the faction they allied with for each war

create view HouseWarHistory as

    select h.name as house, w.name as war, f.name as faction
    from houses h inner join alliedwith a on h.hid = a.hid
    inner join factions f on a.fid = f.fid
    inner join wars w on f.wid = w.wid
    order by house asc;

<table>
<thead>
<tr>
<th>house text</th>
<th>war text</th>
<th>faction text</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baratheon of Dragonstone</td>
<td>Conflict Beyond the Wall</td>
<td>The Night's Watch</td>
</tr>
<tr>
<td>Baratheon of Dragonstone</td>
<td>War of the Five Kings</td>
<td>The King in the Narrow Sea</td>
</tr>
<tr>
<td>Baratheon of King's Landing</td>
<td>War of the Five Kings</td>
<td>The King on the Iron Throne</td>
</tr>
<tr>
<td>Baratheon of King's Landing</td>
<td>Robert's Rebellion</td>
<td>Rebels</td>
</tr>
<tr>
<td>Greyjoy</td>
<td>Robert's Rebellion</td>
<td>Rebels</td>
</tr>
<tr>
<td>Greyjoy</td>
<td>War of the Five Kings</td>
<td>The King of the Isles and the North</td>
</tr>
<tr>
<td>Lannister</td>
<td>Robert's Rebellion</td>
<td>Rebels</td>
</tr>
<tr>
<td>Lannister</td>
<td>War of the Five Kings</td>
<td>The King on the Iron Throne</td>
</tr>
<tr>
<td>Martell</td>
<td>Robert's Rebellion</td>
<td>Royalists</td>
</tr>
<tr>
<td>Stark</td>
<td>War of the Five Kings</td>
<td>The King in the North and the Trident</td>
</tr>
<tr>
<td>Stark</td>
<td>Robert's Rebellion</td>
<td>Rebels</td>
</tr>
<tr>
<td>Targaryen</td>
<td>Robert's Rebellion</td>
<td>Royalists</td>
</tr>
<tr>
<td>Targaryen</td>
<td>War of the Ninepenny Kings</td>
<td>Seven Kingdoms</td>
</tr>
<tr>
<td>Tully</td>
<td>War of the Five Kings</td>
<td>The King in the North and the Trident</td>
</tr>
<tr>
<td>Tully</td>
<td>Robert's Rebellion</td>
<td>Rebels</td>
</tr>
</tbody>
</table>
**Fealty**: Lists each living person who swears fealty to a lord as well as that lord’s name

```sql
create view Fealty as
    select p1.name as person, p2.name as lord
    from people p1 inner join allegiance a  on p1.pid   = a.pid
        inner join houses     h  on a.hid    = h.hid
        inner join people     p2 on h.lordid = p2.pid
    where p1.pid       != p2.pid
        and p1.deathyear is null
    order by lord asc;
```

Sample Output 4: Fealty
1. Query to return the number of wars won by each house

```sql
SELECT h.name AS house, COUNT(h.hid) AS warsWon
FROM houses h
INNER JOIN alliedwith a ON h.hid = a.hid
INNER JOIN factions f ON a.fid = f.fid
WHERE f.wonwar = true
GROUP BY h.name
ORDER BY warsWon DESC;
```

<table>
<thead>
<tr>
<th>House Text</th>
<th>Wars Won</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lannister</td>
<td>2</td>
</tr>
<tr>
<td>Baratheon of King's Landing</td>
<td>2</td>
</tr>
<tr>
<td>Tully</td>
<td>1</td>
</tr>
<tr>
<td>Stark</td>
<td>1</td>
</tr>
<tr>
<td>Greyjoy</td>
<td>1</td>
</tr>
<tr>
<td>Targaryen</td>
<td>1</td>
</tr>
</tbody>
</table>

2. Query to return the length in years of each king’s reign

```sql
SELECT p.name,
       CASE WHEN COALESCE((k.reignendyear - k.reignstartyear), (300 - reignstartyear)) = 0
            THEN 1
       ELSE COALESCE((k.reignendyear - k.reignstartyear), (300 - reignstartyear))
       END AS reignlengthyears
FROM people p
INNER JOIN kings k ON p.pid = k.pid
ORDER BY reignlengthyears DESC;
```

<table>
<thead>
<tr>
<th>Name Text</th>
<th>Reignlengthyears Integer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Robert Baratheon</td>
<td>15</td>
</tr>
<tr>
<td>Joffrey Baratheon</td>
<td>2</td>
</tr>
<tr>
<td>Stannis Baratheon</td>
<td>1</td>
</tr>
<tr>
<td>Robb Stark</td>
<td>1</td>
</tr>
<tr>
<td>Balon Greyjoy</td>
<td>1</td>
</tr>
<tr>
<td>Tommen Baratheon</td>
<td>1</td>
</tr>
<tr>
<td>Euron Greyjoy</td>
<td>1</td>
</tr>
</tbody>
</table>

Sample Output 5

Sample Output 6
3. Query to return the allegiance of each person as well as their assumed aliases

```sql
select p.name, coalesce(a1.name, '(no aliases)') as alias, h.name
from people p
left outer join aliases a1 on p.pid = a1.pid
inner join allegiance a2 on p.pid = a2.pid
inner join houses h on a2.hid = h.hid
order by p.name;
```

Sample Output:

<table>
<thead>
<tr>
<th>Name</th>
<th>Alias Text</th>
<th>Name Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arya Stark</td>
<td>Cat of the Canals</td>
<td>Stark</td>
</tr>
<tr>
<td>Arya Stark</td>
<td>Ary</td>
<td>Stark</td>
</tr>
<tr>
<td>Arya Stark</td>
<td>Nymeria</td>
<td>Stark</td>
</tr>
<tr>
<td>Bran Stark</td>
<td>(no aliases)</td>
<td>Greyjoy</td>
</tr>
<tr>
<td>Brienne of Tarth</td>
<td>Brienne the Beauty</td>
<td>Stark</td>
</tr>
<tr>
<td>Brienne of Tarth</td>
<td>The Maid of Tarth</td>
<td>Tully</td>
</tr>
<tr>
<td>Brienne of Tarth</td>
<td>Brienne the Beauty</td>
<td>Stark</td>
</tr>
<tr>
<td>Catelyn Stark</td>
<td>Lady Stoneheart</td>
<td>Stark</td>
</tr>
<tr>
<td>Catelyn Stark</td>
<td>Lady Stoneheart</td>
<td>Tully</td>
</tr>
<tr>
<td>Cersei Lannister</td>
<td>(no aliases)</td>
<td>Lannister</td>
</tr>
<tr>
<td>Cersei Lannister</td>
<td>(no aliases)</td>
<td>Baratheon of King's Landing</td>
</tr>
<tr>
<td>Daenerys Targaryen</td>
<td>Mother of Dragons</td>
<td>Targaryen</td>
</tr>
<tr>
<td>Daenerys Targaryen</td>
<td>Stormborn</td>
<td>Targaryen</td>
</tr>
<tr>
<td>Davos Seaworth</td>
<td>Onion Knight</td>
<td>Baratheon of Dragonstone</td>
</tr>
<tr>
<td>Haryn Martell</td>
<td>(no aliases)</td>
<td>Martell</td>
</tr>
<tr>
<td>Eddard Stark</td>
<td>(no aliases)</td>
<td>Stark</td>
</tr>
<tr>
<td>Edmure Tully</td>
<td>(no aliases)</td>
<td>Tully</td>
</tr>
<tr>
<td>Euron Greyjoy</td>
<td>Crow's Eye</td>
<td>Greyjoy</td>
</tr>
<tr>
<td>Jaime Lannister</td>
<td>Kingslayer</td>
<td>Lannister</td>
</tr>
<tr>
<td>Joffrey Baratheon</td>
<td>(no aliases)</td>
<td>Baratheon of King's Landing</td>
</tr>
<tr>
<td>Joffrey Baratheon</td>
<td>(no aliases)</td>
<td>Lannister</td>
</tr>
<tr>
<td>Jon Snow</td>
<td>99th Lord Commander of the Night's Watch</td>
<td>Stark</td>
</tr>
<tr>
<td>Oberyn Martell</td>
<td>The Red Viper</td>
<td>Martell</td>
</tr>
<tr>
<td>Robb Stark</td>
<td>The Young Wolf</td>
<td>Stark</td>
</tr>
<tr>
<td>Robert Baratheon</td>
<td>The Umurper</td>
<td>Baratheon of King's Landing</td>
</tr>
<tr>
<td>Sansa Stark</td>
<td>Ailane Stone</td>
<td>Stark</td>
</tr>
<tr>
<td>Sandor Clegane</td>
<td>(no aliases)</td>
<td>Baratheon of Dragonstone</td>
</tr>
<tr>
<td>Theon Greyjoy</td>
<td>Reek</td>
<td>Greyjoy</td>
</tr>
<tr>
<td>Tommen Baratheon</td>
<td>(no aliases)</td>
<td>Baratheon of King's Landing</td>
</tr>
<tr>
<td>Tommen Baratheon</td>
<td>(no aliases)</td>
<td>Lannister</td>
</tr>
<tr>
<td>Tywin Lannister</td>
<td>The Imp</td>
<td>Lannister</td>
</tr>
<tr>
<td>Tyrion Lannister</td>
<td>Halfman</td>
<td>Lannister</td>
</tr>
<tr>
<td>Tywin Lannister</td>
<td>Volo</td>
<td>Lannister</td>
</tr>
<tr>
<td>Tywin Lannister</td>
<td>(no aliases)</td>
<td>Lannister</td>
</tr>
</tbody>
</table>

Sample Output 7
**Stored Procedures**

**percentBattlesWon**: Takes a faction name as an argument and returns the percentage of battles that faction won and whether or not they won the war.

```sql
create or replace function percentBattlesWon(factionName text)
returns table(percent_won numeric, wonwar boolean) as $$
begin
  return query select trunc (  
    (cast(  
      ( select count(c.fid) as battleswon  
        from combatants c inner join factions f on c.fid = f.fid  
        inner join wars     w on w.wid = f.wid  
        where f.name      = factionName  
          and c.wonbattle = true  
      ) as decimal(5,2))
    /  
    ( select count(c.fid) as battlesfought  
      from combatants c inner join factions f on c.fid = f.fid  
      inner join wars     w on w.wid = f.wid  
      where f.name = factionName  
    ) * 100) , 2
  ) as percent_won, factions.wonwar
  from factions
  where name = factionName;
end; $$ language plpgsql;
```

```sql
select percentBattlesWon('The King in the North and the Trident');
```

**Sample Output 8:**
percentBattlesWon

(37.50, f)
**endWar**: Returns a trigger that sets the endyear for a war to the current year if a faction is updated to have won the war and the war is not over (See Triggers, p.23)

```
create or replace function endWar() returns trigger as
$$
declare
-- Unfortunately, the Westerosi calendar is not supported by Postgres, necessitating the following -- variable:
   currentYear integer := 300;
begin
  if new.wonwar = true
      and (select endyear
           from wars
           where wid = new.wid) is null
  then
      update Wars
      set endyear = currentYear
      where wid = new.wid;
  end if;
  return new;
end;
$$
language plpgsql;
```
**endReign**: Returns a trigger that sets the endreignyear of a king to their death year if they are updated to have died and their reign is not currently over (See Triggers, p.24)

```sql
create or replace function endReign() returns trigger as
$$
begin
  if new.deathyear is not null
  and (select reignendyear
       from kings
       where pid = new.pid) is null
  then
    update kings
    set reignendyear = new.deathyear
    where pid = new.pid;
  end if;
  return new;
end;
$$
language plpgsql;
```
**Triggers**

**endWar:** After updating the Factions table, executes the endWar() procedure (See Stored Procedures, p.21)

```sql
create trigger endWar
after update on Factions
for each row
execute procedure endWar();
```

*Sample Output 9: endWar*
**endReign:** After updating the People table, executes the endReign() procedure (See Stored Procedures, p.22)

```sql
create trigger endReign
after update on People
for each row
execute procedure endReign();
```

<table>
<thead>
<tr>
<th>Before Update</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>pid</strong></td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>6</td>
</tr>
</tbody>
</table>

| 6 | Iron King of the Isles and the North | | 299 | |

<table>
<thead>
<tr>
<th>After Update</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>pid</strong></td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>6</td>
</tr>
</tbody>
</table>

| 6 | Iron King of the Isles and the North | | 299 | 300 |

*Sample Output 10: endReign*
Administrator Role: Represents the database administrator who has full access

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

Maester Role: Represents the maesters who work at the Citadel who need to access and change records for their studies and for archival purposes

create role maesters;
revoke all on all tables in schema public from maesters;
grant select on all tables in schema public to maesters;
grant insert on people, kings, nightswatchmen, knights, aliases, houses, allegiance, foughtin, battles, wars, combatants, factions, alliedwith to maesters;
grant update on people, kings, nightswatchmen, knights, aliases, houses, allegiance, foughtin, battles, wars, combatants, factions, alliedwith to maesters;
Visitor Role: Represents an outside visitor to the Citadel who needs to access data but is not allowed to change anything

create role visitors;
revoke all on all tables in schema public from visitors;
grant select on people, kings, nightswatchmen, knights,
    houses, castles, regions, foughtin,
    battles, wars, combatants, factions,
    alliedwith

to visitors;
Implementation Notes

- Postgres does not support the Westerosi calendar including years and dates. For this reason, years are stored as integers and the current year is manually entered when necessary.

- In all instances when definitive historical data is not yet available, the value is left as null. For instance, ongoing wars have a null endyear and living people have a null deathyear.

- All names of people, houses, wars, factions, battles, etc. do not have to be unique. For instance, a house could splinter into warring factions but maintain the same name in both separate houses. For this reason and similar others, names are not used as primary keys in any tables.

- For those not acquainted with Westerosi culture, a “maester” is a scholarly individual comparable to a teacher, doctor, or other learned person. They study at the Citadel and are often assigned to houses to serve as advisors and doctors.
**Known Problems / Future Enhancements**

- Due to the date problem described in Implementation Notes, the current year will have to be updated in the database each year.

- With the current implementation, it is difficult to tell exactly who is related to whom. One can see which people have familial allegiance to the same house, but even then it will not count people who have married into that house. Thus, a future version could improve on the current implementation by taking into account family trees and creating a hierarchy of sorts.

- There are many more tables that could be implemented in future versions of the database. For instance, there could be a “Groups” or “Organizations” table which has subtypes such as “Houses”, “Mercenaries”, or “Religions”. This would allow a greater level of detail showing different people’s allegiances to groups other than just Houses.