Fall 2022 B461 Assignment 1
Relational Model and Intro to SQL

Chinmayee Mundhe, Sachitha Suresh, Muazzam Siddiqui

Released: August 26, 2022
Due: September 08, 2022

1 Introduction

The goals for this assignment are to

1. become familiar with the PostgreSQL system[1];
2. create a relational database and populate it with data;
3. examine the side-effects on the state of the database caused by inserts and deletes in the presence or absence of primary and foreign key constraints;
4. formulate some queries in SQL and evaluate them in PostgreSQL; and

To turn in your assignment, you will need to upload to Canvas the following files zipped into one :

- `assignment1.sql`
- `assignment1.txt`
- `assignment1.pdf`

The `assignment1.sql` contains the necessary SQL statements that solve the problems in this assignment. The `assignment1.sql` file must be such that the AI's can run it in their PostgreSQL environment.

The `assignment1.txt` file contains the results of running your queries.

The `assignment1.pdf` file contains the solutions for theoretical problems.

[1] To solve this assignment, you will need to download and install PostgreSQL (version 12 or higher) on your computer.

For the problems in this assignment we will use the following database schema:[2]

```
Westerosi(wid, wname, wlocation)
House(hname,kingdom)
Skill(skill)
OfHouse(wid, hname, wages)
HouseAllyRegion(hname,region)
WesterosiSkill(wid,skill)
Predecessor(succid,predid)
Knows(wid1,wid2)
```

In this database[3] we maintain a set of Westerosis[4] (`Westerosi`), a set of Houses (`House`), and a set of skills (`Skill`). The `wname` attribute in `Westerosi` is the name of the resident of Westeros.

The `wlocation` attribute in `Westerosi` specifies the area in which the person is currently stationed. The `hname` attribute in `House` is the name of a House in Westeros.

The `kingdom` attribute in `House` is the name of the location wherein the lord of the house resides. The `skill` attribute in `Skill` is the name of a skill possessed by Westerosis.

A Westerosi can be of at most one House. This information is maintained in the `OfHouse` relation. (We permit that a Westerosi does not belong to any House.) The `wages` attribute in `OfHouse` specifies the wages made by the Westerosi.

The `region` attribute in `HouseAllyRegion` indicates a region in which the house has allies. (Houses may have allies in multiple regions.)

A Westerosi can have multiple skills. This information is maintained in the `WesterosiSkill` relation. A skill can be the skill of multiple Westerosis. (A Westerosi may not have any skills, and a skill may have no Westerosis with that skill.)

A pair (s, p) in `Predecessor` indicates that a Westerosi (successor) s has a Westerosi p as one of his or her predecessors. We permit that a successor has multiple predecessors and that a predecessor may be succeeded by multiple successors. (It is possible that a Westerosi has no predecessor and that a Westerosi is not a predecessor.) We further require that a Westerosi and his or her predecessors must belong to the same House.

The relation `Knows` maintains a set of pairs (w_1, w_2) where w_1 and w_2 are wids of Westerosis. The pair (w_1, w_2) indicates that the person with wid w_1

[2] The primary key, which may consist of one or more attributes, of each of these relations is underlined.

[3] The values of the database are inspired by a popular series - Game of Thrones just to make the course a little fun. We in no way bear responsibility for any spoilers or faults in the storyline/theories based on these values. So kindly humor us and have just as fun with making the queries as we do in asking for them!

[4] Residents of Westeros

knows the person with wid w_2. We do not assume that the relation Knows is symmetric: it is possible that (w_1, w_2) is in the relation but that (w_2, w_1) is not.

The domain for the attributes wid, wages, succid, and predid is integer. The domain for all other attributes is text.

We assume the following foreign key constraints:

- wid is a foreign key in OfHouse referencing the primary key wid in Westerosi;
- hname is a foreign key in OfHouse referencing the primary key hname in House;
- hname is a foreign key in HouseAllyRegion referencing the primary key hname in House;
- wid is a foreign key in WesterosiSkill referencing the primary key wid in Westerosi;
- skill is a foreign key in WesterosiSkill referencing the primary key skill in Skill;
- succid is a foreign key in Predecessor referencing the primary key wid in Westerosi; and
- predid is a foreign key in Predecessor referencing the primary key wid in Westerosi;
- wid1 is a foreign key in Knows referencing the primary key wid in Westerosi; and
- wid2 is a foreign key in Knows referencing the primary key wid in Westerosi

The file data.sql contains the data supplied for this assignment.

2 Database creation and impact of constraints on insert and delete statements.

Create a database in PostgreSQL that stores the data provided in the `data.sql` file. Make sure to specify primary and foreign keys.

1. Create a new table named 'WhiteWalker' with primary key as 'ranking' of type 'integer' such as 1, 2,... along with foreign keys as 'wid' and 'skill' from table Westerosi and Skill respectively. Include 'nkname' of type 'text' and 'kills' of type 'integer' as the other attributes for the table. The key attribute should follow 'NOT NULL' Constraint rule. Insert 3 or more tuple values to populate the table. Make sure that at least 2 of the WhiteWalkers have more than 5 kills

2. Find all the details of the WhiteWalkers who have more than 5 kills. (Make sure that at least 2 of the WhiteWalkers have more than 5 kills.)

3. Find each (wid, wname, wlocation) tuple such that they only belong to the house of 'Starks'.

4. With the help of set operation EXCEPT formulate the query, find the wid of the Westerosi who don't have 'Archery' or 'Swordsmanship' as their skill.

5. Provide 4 conceptually different examples that illustrate how the presence or absence of primary and foreign keys affect insert and deletes in these relations. To solve this problem, you will need to experiment with the relation schemas and instances for this assignment. For example, you should consider altering primary keys and foreign key constraints and then consider various sequences of insert and delete operations. You may need to change some of the relation instances to observe the desired effects. Certain inserts and deletes should succeed but other should generate error conditions. (Consider the lecture notes about keys, foreign keys, and inserts and deletes as a guide to solve this problem.)

3 Formulating queries in SQL

For this assignment, you are required to use tuple variables in your SQL statements. For example, in formulating the query "Find the pid and pname of each person who lives in Bloomington" you should write the query

```
SELECT  p.pid, p.pname
FROM    Person p
WHERE   p.city = 'Bloomington'
```

rather than

```
SELECT  pid, pname
FROM    Person
WHERE   city = 'Bloomington'
```

Write SQL statements for the following queries. Make sure that each of your queries returns a **set** but not a bag. In other words, make appropriate use of the `DISTINCT` clause where necessary.

You **can not** use the SQL JOIN[5] operations or SQL aggregate functions such as `COUNT`, `SUM`, `MAX`, `MIN`, etc in your solutions.

6. Find the ID, name and the house that they belong to of each Westerosi who (a) belongs to a house that has allies in Casterly Rock (b) has Archery as a skill and (c) has wages between 50000 and 75000

7. Find each house that has at least 2 Westerosis belonging to it.

8. Find the ID, name and location of each Westerosi who has at least one predecessor. **NOTE:** To get full credit for this question please use Subquery.

   For the following questions please ensure at least one use of Set operations:

9. Find the wids of the westerosis such that each has either Archery or Politics in their skill set and they either belong to House Stark or House Baratheon.

10. Find the wids of each westerosi who either has **no** predecessors or **not have both** a House nor a skill. **Extra Credit:** To get extra credit for this question please use only set operations and do not use where condition.

[5] A JOIN operation is different than a join condition; You can use Join condition.