Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

CSU3401 Information Management 2
SQL Project

Merlin Prasad Std #19333557

March 21,2022

Section A: Description of Database Application area and ER Model

1. Application Description

This is the database application for a fictional private nursing home in Ireland. The nursing home provides residential care for both elderly and disabled people with a great range of facilities. The residents are taken care of by a skilled team of staff members. To represent the nursing home I used the following entities : residents , staff , wards , medication and recreational activities.

I used the following assumptions for modelling my ER diagram and database .

Staff

The nursing home employs multiple staff from different positions. Each staff member works in a specific ward of the nursing home .Staff members can be medical professionals as well as non medical professionals . As this is a nursing home in Ireland each staff member has a PPSN which is a unique form of identification. Staff hold multiple different degrees and certifications. Staff are also all paid at least minimum wage .

Ward

Each ward also contains multiple different residents. The ward has a unique identification number associated with it.The nursing home contains exactly 5 wards in it

Resident

Each resident stays in exactly one ward of the nursing home. Wards are mixed gender and residents may move from one ward to another during the course of their stay. On the day the resident is admitted to the nursing home they are supplied with a unique resident ID.Residents must at least be 18 years of age to be able to be admitted into the nursing home. Residents may share the same date of birth as well as the date they were admitted to the nursing home.

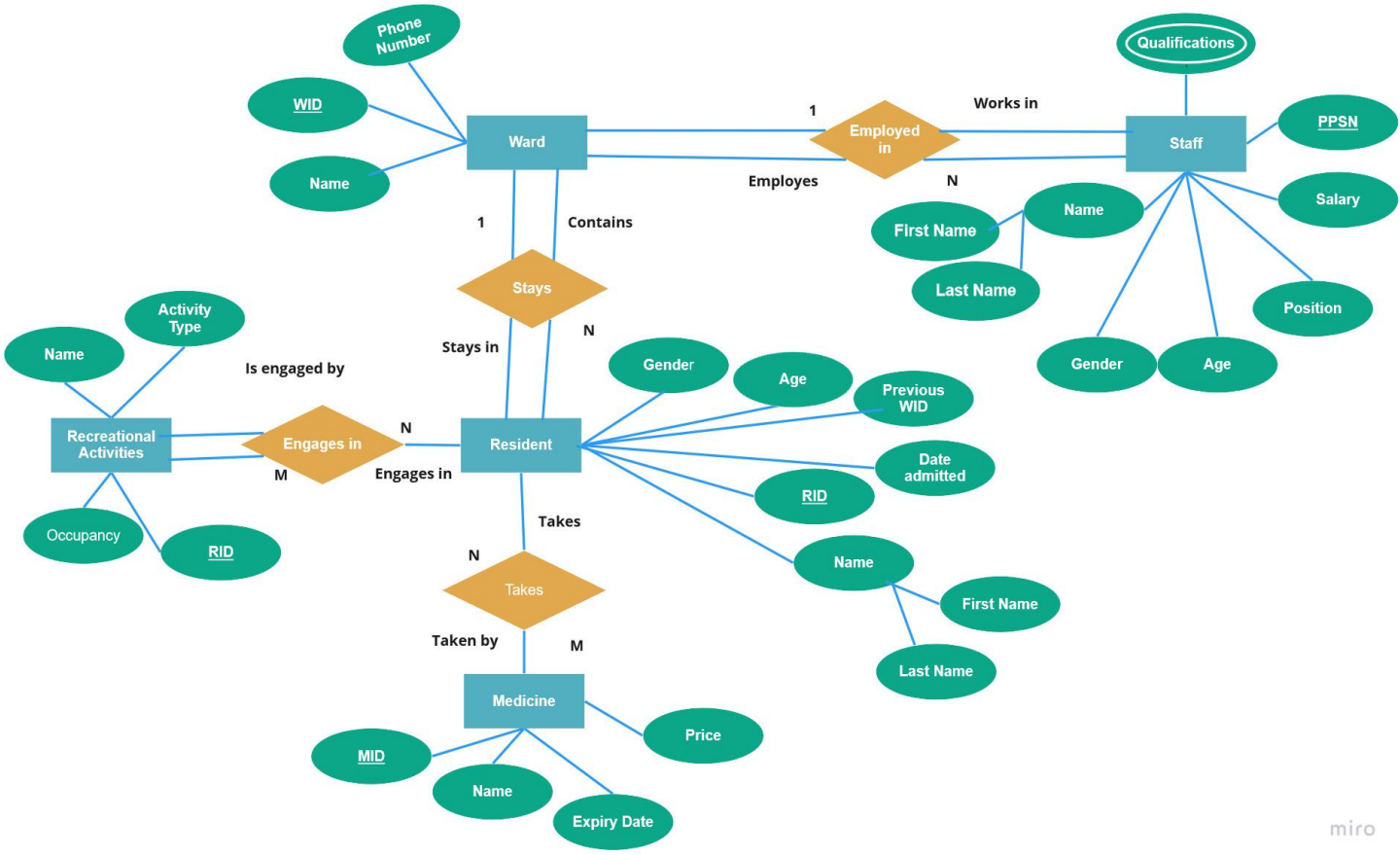
Recreational activities

Residents are able to engage in many different recreational activities provided by the nursing home. Participating in an activity is optional and residents can take part in multiple different activities if they wish. The activities offered by the nursing home all have a unique activity ID associated with it.

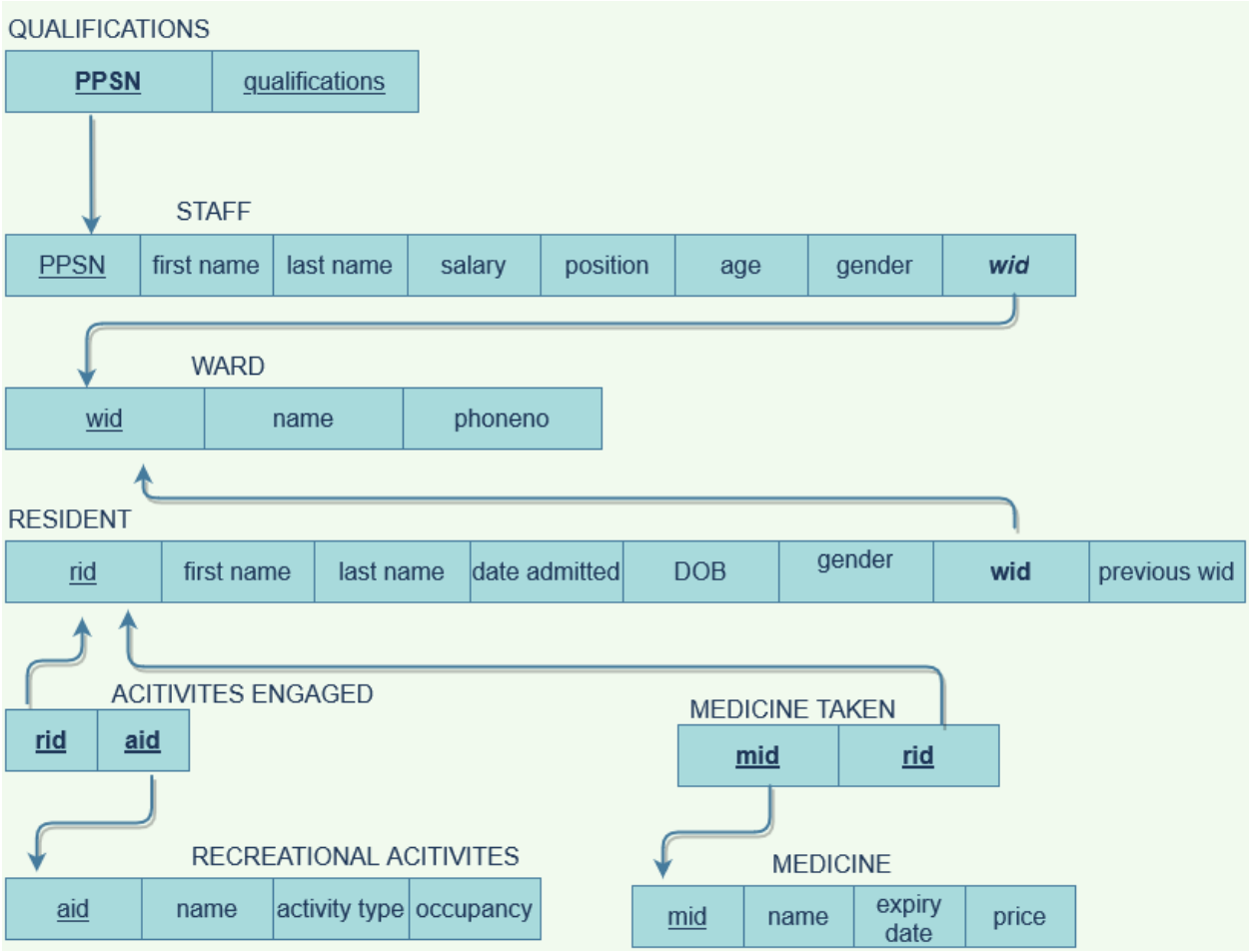
Medicine

Most residents in the nursing home take medication as well. All medications have a unique medication ID . Some residents require more medication than others . For safety reasons none of the medication kept in the nursing home is past its expiration date . Keeping track of the price of the medicine is also important for future reference.

2. Entity Relationship Diagram.



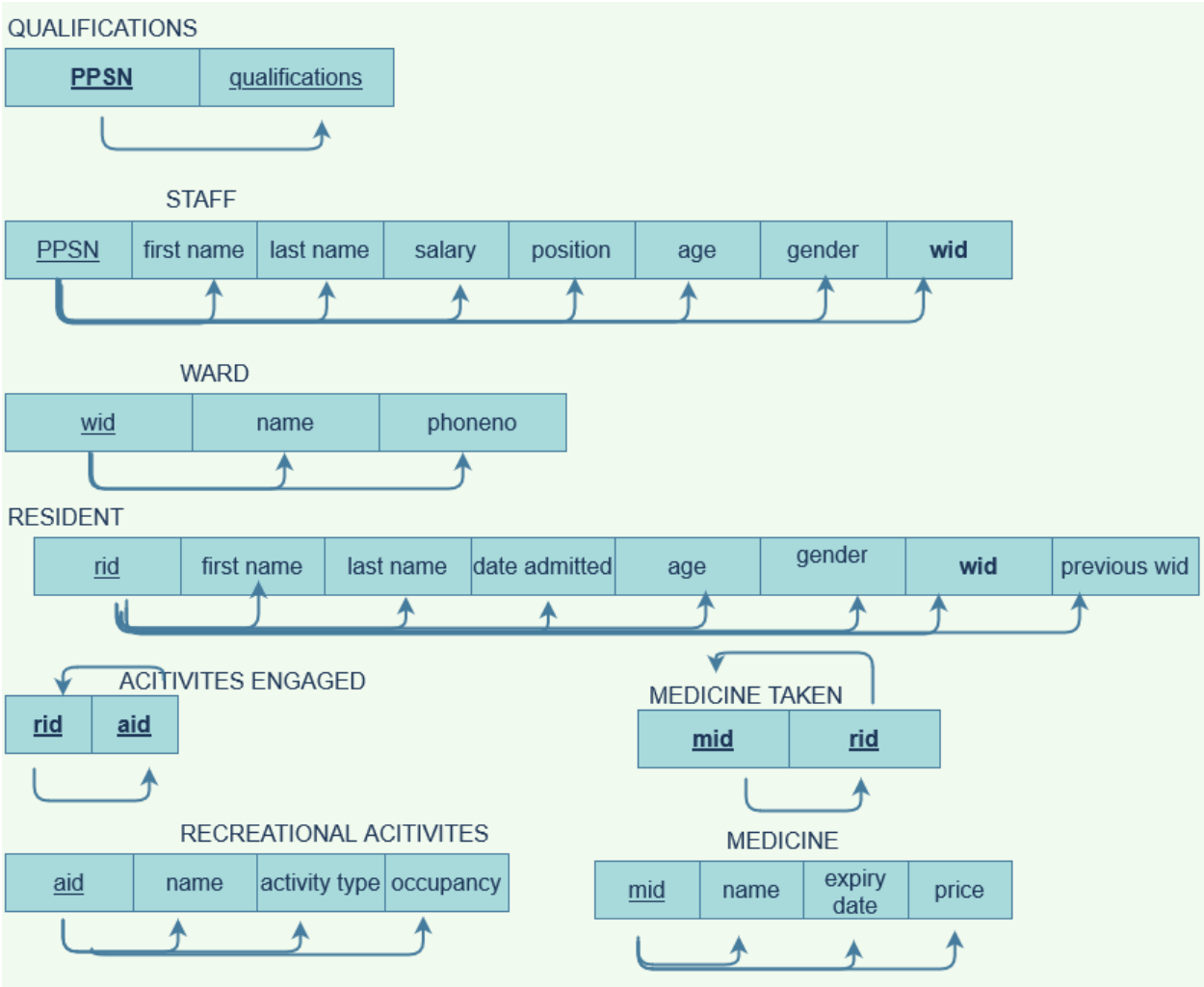
3. Mapping to Relational Schema



Primary keys are <u>underlined</u>

Foreign keys are in **bold**

4. Functional Dependency Diagrams (for proposed relations)



Note:

Primary keys are <u>underlined</u>

Foreign keys are in **bold**

Section B Explanation of data and SQL Code:

5. Explanation of one the SQL code for creating one of your database tables (including any constraints)

```sql
CREATE TABLE `staff` (
    `PPSN` VARCHAR(9) NOT NULL,
     `first_name` VARCHAR(45) NOT NULL,
    `last_name` VARCHAR(45)  NULL,
    `salary` INT NULL,
    `position` VARCHAR(45) NOT NULL,
    `age` INT NOT NULL,
    `gender` VARCHAR(45) NOT NULL,
    `wid` INT ,
    CHECK (wid >= 1 AND wid <= 5 ),
    PRIMARY KEY (`PPSN`),
    FOREIGN KEY (`wid`) REFERENCES `ward`(`wid`) );
```

This is the code that creates the staff table for the staff entity.

The primary key for the staff table is PPSN as this is a nursing home in Ireland.so all PPSN are unique identifiers. PPSN are also always a maximum of 9 characters.

WID stands for ward ID and as this nursing home only has 5 wards the check ensures all staff should only be assigned to one of these wards. WID is the foreign key that links the staff table to the ward table.

Name is a composite attribute that contains first name and last name in the staff entity

Other than foreign and primary keys, age, first name and position and gender are not allowed to be null in the database as these are all crucial pieces of information. Salary is allowed to be null here because a trigger in the database will later ensure all staff are being paid at least minimum wage.

6. Explanation and SQL Code for any Altering tables

```
ALTER TABLE resident
ADD previous_wid  INT NULL
AFTER wid ;
```

This table adds a new column of previous wid of a resident in the ward. This will keep track of the last ward a resident stayed at if there is one , otherwise it is null. This is to help keep track of resident information easily

7. Explanation and SQL Code for any Trigger operations

```
delimiter $$
CREATE TRIGGER resident_age
BEFORE INSERT
ON resident
FOR EACH ROW
IF NEW.age < 18 THEN
SIGNAL SQLSTATE '50001' SET MESSAGE_TEXT = 'Residents must be 18 or older.';
END IF; $$
delimiter ;
```

Residents in the nursing home must be at least 18 years old so this trigger operation will activate if any new resident is added to the resident table that is below the minimum age and inform them about the issue.

```
delimiter $$
CREATE TRIGGER minimum_salary
BEFORE INSERT
ON staff
FOR EACH ROW
IF NEW.salary < 20685
THEN SET NEW.salary = 20685 ;
END IF; $$
delimiter ;
```

This trigger makes sure all staff in the database are being paid at least minimum wage by checking that all new staff inserted are paid at least 20685 euros per year (minimum wage per annum in Ireland in 2022)

```
delimiter $$
CREATE TRIGGER old_wid
BEFORE UPDATE
ON resident
FOR EACH ROW
BEGIN
IF NEW.wid <> OLD.wid THEN
 SET NEW.previous_wid = OLD.wid ;
END IF;
END $$
delimiter ;
```

This trigger automatically updates the previous_wid column of a resident whenever a resident is moved from one ward to another. It sets the previous_wid as the old.wid before the update occurred. This helps facilitate the movement of residents from one ward to another.

8. Explanation and SQL Code for any Creation of Views.

```sql
CREATE VIEW ward_occupancy AS
SELECT ward.wid as "ward id", ward.name AS "ward name", COUNT(resident.rid) "no of residents"
FROM resident
JOIN ward ON ward.wid = resident.wid
GROUP BY ward.wid;
```

	ward id	ward name	no of residents
▶	1	Seton	2
	2	Nerses	2
	3	Jacques	1
	4	Vincenz	2
	5	Elizabeth	4

This view shows the occupancy of all the wards by joining the ward and resident tables and using the count function to count up all the resident id in each of the wards . The occupancy iis grouped by the ward ids . Aliasing is also used to give meaningful column names such as ward id over wid and ward name over name so anyone looking at this table can understand what the columns represent.

```
CREATE VIEW resident1_meds AS
SELECT medicine.name, medicine.mid, medicine.expiry_date, price
FROM medicine
JOIN medicine_taken ON medicine_taken.mid = medicine.mid
JOIN resident ON medicine_taken.rid = resident.rid
WHERE medicine_taken.rid =
    (SELECT medicine_taken.rid
     WHERE resident.rid = "1") ;
```

Result Grid | Filter Rows: | Export:

name	mid	expiry_date	price
azithromycin	31	2025-06-07	121
sulfonylureas	51	2022-12-09	150

This view shows all the medication resident with id of 1 is taking .Using two joins combines the medicine taken with the resident and medicine tables. This allows for querying this M-N relationship and finding all the medication this individual takes . It is important to the nursing home to easily have access to the medical information of their residents and ensure they are being prescribed properly.

9. Explanation and SQL Code for one of your commands to Populate a Tables

```sql
INSERT INTO `nursing_home`.`medicine` (`mid`, `name`, `expiry_date`, `price`) VALUES ('51', 'sulfonylureas', '2022/12/09', '150');
INSERT INTO `nursing_home`.`medicine` (`mid`, `name`, `expiry_date`, `price`) VALUES ('61', 'plendil', '2024/10/5', '70');
INSERT INTO `nursing_home`.`medicine` (`mid`, `name`, `expiry_date`, `price`) VALUES ('11', 'hydrocodone', '2024/10/3', '44');
INSERT INTO `nursing_home`.`medicine` (`mid`, `name`, `expiry_date`, `price`) VALUES ('21', 'simvastatin', '2023/1/3', '32');
INSERT INTO `nursing_home`.`medicine` (`mid`, `name`, `expiry_date`, `price`) VALUES ('31', 'azithromycin', '2025/6/7', '121');
INSERT INTO `nursing_home`.`medicine` (`mid`, `name`, `expiry_date`, `price`) VALUES ('41', 'paracetomol', '2023/5/5', '11');
INSERT INTO `nursing_home`.`medicine` (`mid`, `name`, `expiry_date`, `price`) VALUES ('71', 'piriton', '2022/3/5', '14');
```

mid	name	expiry_date	price
11	hydrocodone	2024-10-03	44
21	simvastatin	2023-01-03	32
31	azithromycin	2025-06-07	121
41	paracetomol	2023-05-05	11
51	sulfonylureas	2022-12-09	150
61	plendil	2024-10-05	70
71	piriton	2022-03-05	14
NULL	NULL	NULL	NULL

This is the SQL code that populates the medicine table. MID stands for medicine ID and it is the primary key for this table . It is important that the nursing home knows when the expiry date for all the medicine they have is as well as how much it cost to buy for future reference. Mid value also maps the medicine table to the medicine taken relational table so that the nursing home can keep track of what medicine each of the residents take.

10. Explanation and example SQL Code for retrieving information from the database (including any use of Joins and use of functions)

```
SELECT *
FROM staff
WHERE salary >
    (SELECT AVG(salary)
     FROM staff );
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

PPSN	first_name	last_name	salary	position	age	gender	wid
0896326Y	Lisa	Simpson	68846	Director	33	Female	1
1253579K	Nicholas	Riviera	60833	Doctor	46	Male	3
1453679T	Helen	Lovejoy	55469	CNM	48	Female	5
1457266H	Julius	Hibbert	60900	Doctor	59	Male	1

This query selects all the staff in the nursing home that are being paid more than average by using the AVG function.

```
SELECT resident.first_name, resident.last_name, resident.rid
FROM resident
JOIN activities_engaged ON activities_engaged.rid = resident.rid
JOIN recreational_activities ON activities_engaged.aid = recreational_activities.aid
WHERE activities_engaged.aid =
    (SELECT activities_engaged.aid
     WHERE recreational_activities.aid = "8") ;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

first_name	last_name	rid
Hubert	Farnsworth	3
Jasper	Beardly	4
Charles	Burns	7

This nested query finds all the residents that are taking part in activity 8 which is knitting. To do this two joins are used to combine the residents table with the activities engaged table and recreational activities table . This is because residents and recreational activities are a M-N relationship. I have selected to just see each resident's first name, last name and resident id. This query allows the nursing home to see how many residents are attending a specific activity.

11. Explanation and SQL Code for any Security commands (roles & permissions)

```
DROP ROLE IF EXISTS hr_dir ;
CREATE ROLE hr_dir;

DROP ROLE IF EXISTS hr_mgr ;
CREATE ROLE hr_mgr;
```

This code creates two roles in the database: a hr director (hr_dir) and a hr manager (hr_mgr). If those roles exist already then they are dropped and recreated.

```
GRANT CREATE,INSERT,UPDATE,SELECT
ON staff
TO hr_dir
WITH GRANT OPTION;
```

The HR director is given the permission to create,insert ,update and select on the staff relational table . They are also trusted to pass on these permissions to anyone else.

```
CREATE VIEW staff_restricted AS
SELECT first_name,age,position,wid
FROM staff ;
```

A view of the staff table is created that omits sensitive information such as Salary and PPSN.

```
GRANT SELECT,UPDATE,DELETE
ON staff_restricted
TO hr_mgr ;
```

Access to select,update and delete on this view of the staff table is then granted to the hr manager.

12. Explanation of and Additional SQL Features of your choice

Procedures

```
-- stored procedures parameterised
delimiter $$
DROP PROCEDURE IF EXISTS get_residents_ages;
CREATE PROCEDURE get_residents_ages(IN resident_age INT )
BEGIN
    SELECT rid,last_name,age
        FROM resident
        WHERE age >= resident_age
        ORDER BY age , last_name ;
END $$
delimiter ;
CALL get_residents_ages("90");
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

rid	last_name	age
4	Beardly	90
3	Farnsworth	92

This is a stored parameterised procedure that gets all the residents above a supplied age. It is ordered by age and last name hierarchy in descending order. For example supplying 90 in my database gets the residents with rid 4 and 3 as they are the only residents about 90 years of age currently in the nursing home.

Ranked partition

```
SELECT wid,rid,last_name,age, rank()
OVER (partition by wid order by age DESC )
AS "seniority in ward" FROM resident ;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

wid	rid	last_name	age	seniority in ward
1	6	Bouvier	70	1
1	5	Moleman	65	2
2	1	Fry	87	1
2	8	Skinner	87	1
3	7	Burns	80	1
4	3	Farnsworth	92	1
4	9	Simpson	73	2
5	4	Beardly	90	1
5	2	Bagge	76	2
5	10	Miller	76	2
5	11	Abernathy	66	4

Using a ranked partition I was able to categorise all the residents in each ward in order of seniority. The rank function is used to rank the residents by age and the partition splits up the ranking on each ward. This way you can track who is the oldest ,second oldest etc in each of the wards. In the case where both residents have the same age they are given the same rank.

[Parameterised function ,Status and Union]

```
DROP FUNCTION IF EXISTS years_since_date;
delimiter $$
CREATE FUNCTION years_since_date(old_date date ) RETURNS INT DETERMINISTIC
BEGIN
    RETURN year(current_date()) - year(old_date);
END $$
delimiter ;
```

```
-- union of old and new residents
SELECT rid,date_admitted,years_since_date(date_admitted) AS "years at nursing home" , last_name,  "New" AS status
FROM resident
WHERE date_admitted >= "2022-01-01"
UNION
SELECT rid,date_admitted,years_since_date(date_admitted) AS "years at nursing home" ,last_name, "Old" AS status
FROM resident
WHERE date_admitted <= "2022-01-01";
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

rid	date_admitted	years at nursing home	last_name	status
10	2022-01-02	0	Miller	New
11	2022-01-13	0	Abernathy	New
1	2016-12-21	6	Fry	Old
2	2017-07-30	5	Bagge	Old
3	2017-07-30	5	Farnsworth	Old
4	2018-04-17	4	Beardly	Old
5	2018-10-19	4	Moleman	Old
6	2019-07-04	3	Bouvier	Old

Result 27

This is a query of how recently a resident has been admitted to the nursing home. Residents admitted after January 2022 are classified as New and any before that are classified as Old. This can help administrators make sure residents are settling in the nursing home well. Aliasing is also used to give useful column names such as years at the nursing home.

Notice also the function years_since_date. This function takes in a date as a parameter and calculates the number of years that have passed since that date to the current date. By supplying the date the residents were admitted to the nursing home it can calculate how many years each resident has been living there.

Using union on both of these tables I create a new table that contains all of this data

Wildcards

```
CREATE VIEW nurses_employed AS
 SELECT ward.wid AS "ward id", ward.name AS "ward name",  staff.last_name,age, staff.position
 FROM ward
 JOIN staff
 ON ward.wid = staff.wid
 WHERE position LIKE "%Nurse%";
```

	ward id	ward name	last_name	age	position
▶	3	Jacques	Flanders	23	Staff Nurse
	4	Vincenz	Bev	48	Senior Staff Nurse
	5	Elizabeth	Krabappel	58	Staff Nurse

This view shows all the staff employed at the nursing home who are nurses. The Nurse wildcard means all staff with the word nurse in the position is found, which in this case are the Staff Nurse and Senior Staff Nurse. I have selected to only see the ward ID , name , last name and position of these staff. Aliasing is also used to give meaningful column names such as ward id instead of wid.

13. Glossary

- CNM - Clinical nurse manager
- WID - ward identity
- RID - resident identity
- AID - recreational activity identity
- PPSN - personal public service number