



SECTION 1

Creating all Tables in the Schema

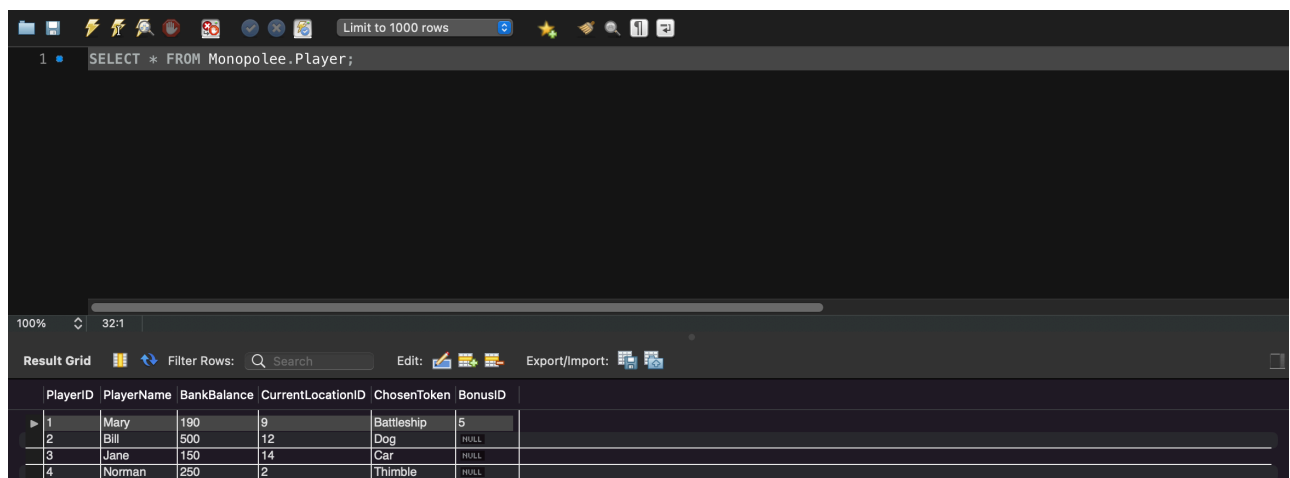
The code for creating the tables is in the link below

[SQL Query to create tables](#)

Populating All Tables in the Schema

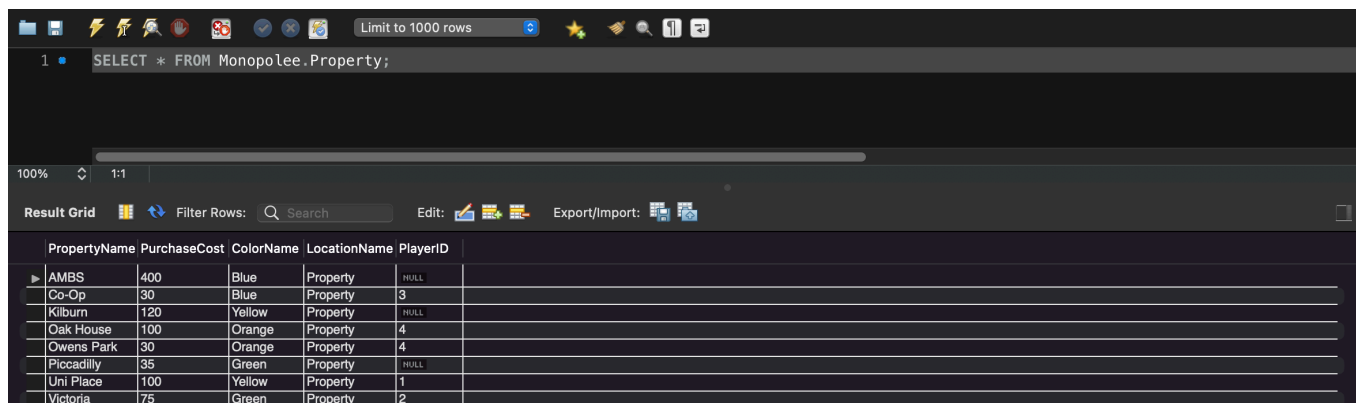
The code for populating the tables is in the link below

[SQL Query to populate tables](#)



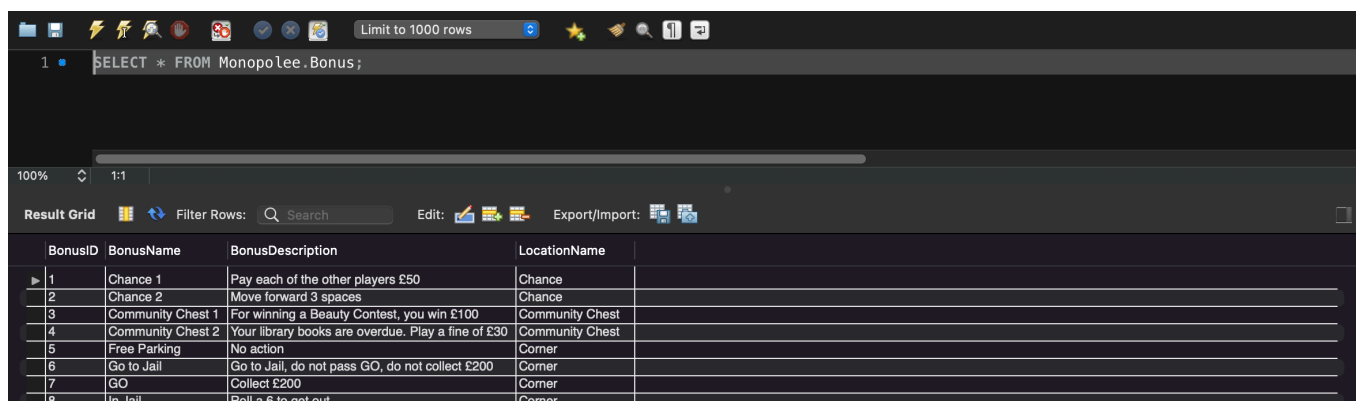
PlayerID	PlayerName	BankBalance	CurrentLocationID	ChosenToken	BonusID
1	Mary	190	9	Battleship	5
2	Bill	500	12	Dog	NULL
3	Jane	150	14	Car	NULL
4	Norman	250	2	Thimble	NULL

Figure 1.1: Screenshot of Player Table showing initial state of the game



PropertyName	PurchaseCost	ColorName	LocationName	PlayerID
AMBS	400	Blue	Property	NULL
Co-Op	30	Blue	Property	3
Kilburn	120	Yellow	Property	NULL
Oak House	100	Orange	Property	4
Owens Park	30	Orange	Property	4
Piccadilly	35	Green	Property	NULL
Uni Place	100	Yellow	Property	1
Victoria	75	Green	Property	2

Figure 1.2: Screenshot of Property Table showing initial state of the game



BonusID	BonusName	BonusDescription	LocationName
1	Chance 1	Pay each of the other players £50	Chance
2	Chance 2	Move forward 3 spaces	Chance
3	Community Chest 1	For winning a Beauty Contest, you win £100	Community Chest
4	Community Chest 2	Your library books are overdue. Play a fine of £30	Community Chest
5	Free Parking	No action	Corner
6	Go to Jail	Go to Jail, do not pass GO, do not collect £200	Corner
7	GO	Collect £200	Corner
8	In Jail	Roll a 6 to get out	Corner

Figure 1.3: Screenshot of Bonus Table

TokenName	
Battleship	
Boot	
Car	
Dog	
Thimble	
Top hat	

Figure 1.4: Screenshot of Token Table

LocationName	
Chance	
Community Chest	
Corner	
Property	

Figure 1.5: Screenshot of Location Table

ColorName	
Blue	
Green	
Orange	
Yellow	

Figure 1.6: Screenshot of Colour Table

Board_LocationID	Board_LocationName	LocationName
1	GO	Corner
2	Kilburn	Property
3	Chance 1	Chance
4	Uni Place	Property
5	In Jail	Corner
6	Victoria	Property
7	Community Chest 1	Community Chest
8	Piccadilly	Property
9	Free Parking	Corner
10	Oak House	Property
11	Chance 2	Chance
12	Owens Park	Property
13	Go to Jail	Corner
14	AMBS	Property
15	Community Chest 2	Community Chest
16	Co-Op	Property

Figure 1.7: Screenshot of Board_Location Table

For the

Audit Trail Table, a trigger was created to automatically insert into it once there's an update to the player table (implying that the player has taken a turn). See link below

[SQL Query to create Audit Trail](#)

SECTION 2

Creating Procedure 'LocationfromDiceRoll'

[SQL Command for Stored Procedure 'LocationfromDiceRoll'](#)

The SQL command creates a procedure **LocationfromDiceRoll** which returns the location of the player after rolling a die. The command takes in three parameters, one of which is optional (Current_Board_Location).

The first section of this procedure calls another procedure to get the initial state of the game (CurrentLocationID) of the player from the Player table and uses that to check if the player is in Jail.

If the Player is in jail then a message is returned to proceed to the next action which is to call another procedure as seen below;

Creating Procedure 'ActionifInJail'

[SQL Command for Stored Procedure 'ActionifInJail'](#)

The procedure 'ActionifInJail' checks if the Player who is currently 'In Jail' rolled a 6 or not. If the Player did not roll a 6 then they are stuck in jail but if they did then they proceed to the next action, which is to roll the dice a second time. This calls for another procedure below;

Creating Procedure 'SecondRollForJail'

[SQL Command for Stored Procedure 'SecondRollForJail'](#)

The procedure 'SecondRollForJail' is very similar to the 'LocationfromDiceRoll' procedure. The only difference is that the former was created specifically to get the new location of Players after release from Jail.

The next section of the **LocationfromDiceRoll** procedure checks if the Player rolled a 6.

If they did, their new position is calculated from their old position (Initial State) and they are asked to roll the dice again by calling another procedure 'SecondRoll' which is again similar to the **LocationfromDiceRoll** procedure. The reason for calling separate procedures when the player must roll the dice again is that a different die number is required.

If they didn't roll a 6 then their new position is calculated and we move on to apply the rules of the game.

Creating Procedure 'Apply_Rules'

[SQL Command for Stored Procedure 'Apply_Rules'](#)

The stored procedure **Apply_Rules** basically applies the rules of the monopoly game depending on where the player lands after rolling a die. The procedure takes in three parameters (Current_Board_Location, PlayerName, Dice_Number) of which the Current_Board_Location is initially optional. It calls the **LocationfromDiceRoll** procedure and stores the variables we would require to set the conditions for the rules, from the **LocationfromDiceRoll** into its own declared variables. Then the 'Current_Board_Location' parameter is set to a declared variable (new_position_name_) which gets its value from the **LocationfromDiceRoll** call.

All other declared variables of this procedure get their value from procedures **PlayerDetails** and **TableVariables** which were created basically to store certain values we would need to set conditions for the Rules.

Please see below;

[SQL Command for Stored Procedure 'PlayerDetails'](#)

[SQL Command for Stored Procedure 'TableVariables'](#)

There are more stored procedures 'Apply_Rules_Jail' and 'Apply_Rules_SecondRoll' which are very similar to the 'Apply_Rules' procedure. The only difference is that the **Apply_Rules_Jail** is used to apply the game rules when the player plays the second time due to him rolling a '6' while trying to get out of jail and the **Apply_Rules_SecondRoll** is used to apply the game rules just when the player plays the second time due to him rolling a '6'.

[SQL Command for Stored Procedure 'Apply_Rules_Jail'](#)

[SQL Command for Stored Procedure 'Apply_Rules_SecondRoll'](#)

SECTION 3:

SQL Commands for each gameplay move GamePlay Round 1

Jane rolls a 3;

```
Call LocationfromDiceRoll(NULL,'Jane',3);  
Call Apply_Rules(NULL,'Jane',3);  
SELECT * FROM Player;
```

	PlayerID	PlayerName	BankBalance	CurrentLocationID	ChosenToken	BonusID	
▶	1	Mary	190	9	Battleship	5	
	2	Bill	500	12	Dog	NULL	
	3	Jane	350	1	Car	7	
	4	Norman	250	2	Thimble	NULL	

Figure 3.1: Screenshot of Player Table after applying rule for 'GO'

Jane landed on GO, therefore her new BankBalance has been increased by £200. Her location has been updated to 1, because she's currently at 'GO' and the Bonus she received at that location has been updated.

Norman rolls a 1;

```
Call LocationfromDiceRoll(NULL,'Norman',1);  
Call Apply_Rules(NULL,'Norman',1);  
SELECT * FROM Player;
```

	PlayerID	PlayerName	BankBalance	CurrentLocationID	ChosenToken	BonusID	
▶	1	Mary	240	9	Battleship	5	
	2	Bill	550	12	Dog	NULL	
	3	Jane	400	1	Car	7	
	4	Norman	200	3	Thimble	1	

Figure 3.2: Screenshot of Player Table after applying rule for 'Chance 1'

£50 has been added to all players' BankBalance and £50 has been deducted from Normans BankBalance. His Location has been updated to 3 which is 'Chance 1' on the board and the Bonus he received at that location has also been updated.

Mary rolls a 4;

```
Call LocationfromDiceRoll(NULL,'Mary',4);  
Call Apply_Rules(NULL,'Mary',4);  
SELECT * FROM Player;
```

	PlayerID	PlayerName	BankBalance	CurrentLocationID	ChosenToken	BonusID	
▶	1	Mary	240	5	Battleship	6	
	2	Bill	550	12	Dog	NULL	
	3	Jane	400	1	Car	7	
	4	Norman	200	3	Thimble	1	

Figure 3.3: Screenshot of Player Table after applying rule for 'Go to Jail'

Mary landed on 'Go to Jail', therefore her location is now 'In Jail'. Her BonusID was also changed to 6.

Bill rolls a 2;

```
Call LocationfromDiceRoll(NULL,'Bill',2);
Call Apply_Rules(NULL,'Bill',2);
SELECT * FROM Player;
```

PlayerID	PlayerName	BankBalance	CurrentLocationID	ChosenToken	BonusID
1	Mary	240	5	Battleship	6
2	Bill	150	14	Dog	NULL
3	Jane	400	1	Car	7
4	Norman	200	3	Thimble	1

Figure 3.4.1: Screenshot of Player Table after applying rule for Property location

Bill landed on AMBS which is a Property that is not owned by any player. The cost of the property has been deducted from his BankBalance and his location updated. He received no Bonus at this location.

```
SELECT * FROM Property;
```

PropertyName	PurchaseCost	ColorName	LocationName	PlayerID
AMBS	400	Blue	Property	2
Co-Op	30	Blue	Property	3
Kilburn	120	Yellow	Property	NULL
Oak House	100	Orange	Property	4
Owens Park	30	Orange	Property	4
Piccadilly	35	Green	Property	NULL
Uni Place	100	Yellow	Property	1
Victoria	75	Green	Property	2

Figure 3.4.2: Screenshot of Property Table after applying rule for Property location

The PlayerID of AMBS has also been updated to 2 (which is Bill's PlayerID) on the property table.

After Gameplay Round 1, the Audit_Trail table is updated to get the PlayerTurn and GameRound with the following command;

```
UPDATE Audit_Trail
SET Audit_Trail.PlayerTurn = CASE WHEN ID >=1 AND ID <=4 THEN 1 ELSE 2 END,
Audit_Trail.GameRound = CASE WHEN ID >=1 AND ID <=4 THEN 1 ELSE 2 END;
SELECT * FROM Audit_Trail;
```

Then the gameView was created using the following command;

```
CREATE OR REPLACE VIEW Gamewiew AS
SELECT Trail.PlayerID,Trail.PlayerName, Trail.CurrentPosition, Trail.PlayerBankBalance,
Trail.PlayerTurn,Trail.GameRound
FROM Audit_Trail AS Trail;

SELECT * FROM Gamewiew;
```

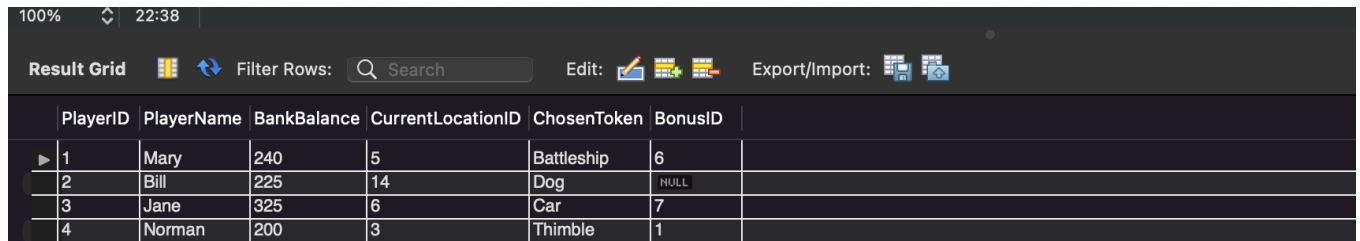
PlayerID	PlayerName	CurrentPosition	PlayerBankBalance	PlayerTurn	GameRound
3	Jane	GO	350	1	1
4	Norman	Chance 1	200	1	1
1	Mary	In Jail	240	1	1
2	Bill	AMBS	150	1	1

Figure 3.5: Screenshot of GameView View after GameRound 1

GamePlay Round 2

Jane rolls a 5;

```
Call LocationfromDiceRoll(NULL,'Jane',5);  
Call Apply_Rules(NULL,'Jane',5);  
SELECT * FROM Player;
```



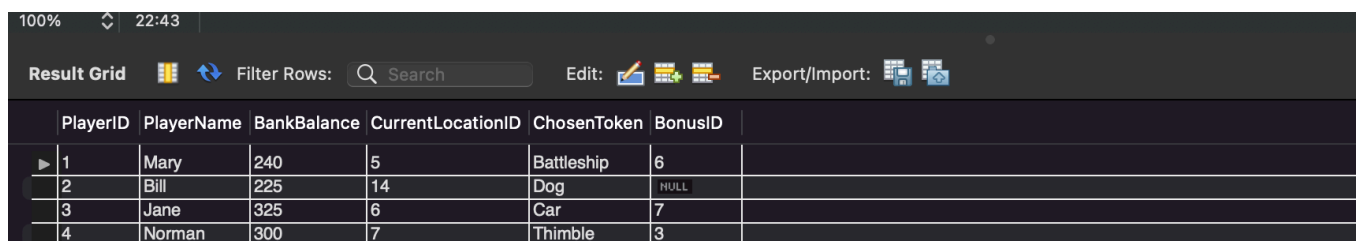
	PlayerID	PlayerName	BankBalance	CurrentLocationID	ChosenToken	BonusID
▶	1	Mary	240	5	Battleship	6
	2	Bill	225	14	Dog	NULL
	3	Jane	325	6	Car	7
	4	Norman	200	3	Thimble	1

Figure 3.6: Screenshot of Player Table after applying rule for Property Location

Jane landed on ‘Victoria’, a property that already has an owner. Her location has been updated to 6 and the rental cost (£75) of the property has been deducted from her BankBalance. Also, Bill who is the owner of the property has received the rent of £75. She didn’t receive a Bonus at this location.

Norman rolls a 4;

```
Call LocationfromDiceRoll(NULL,'Norman',4);  
Call Apply_Rules(NULL,'Norman',4);  
SELECT * FROM Player;
```



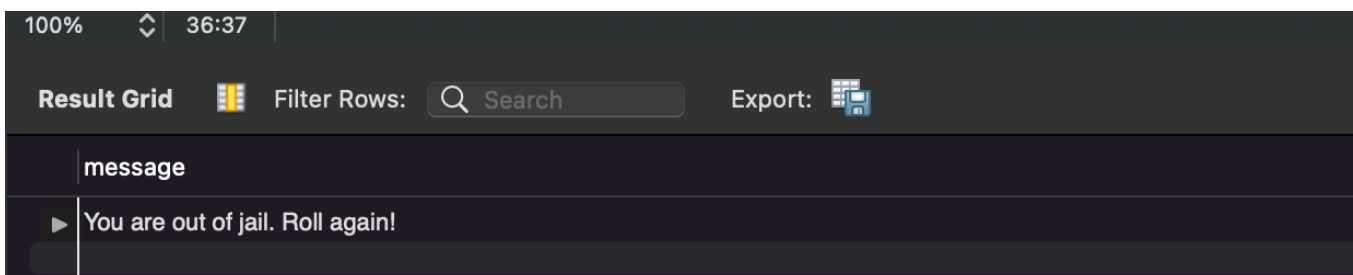
	PlayerID	PlayerName	BankBalance	CurrentLocationID	ChosenToken	BonusID
▶	1	Mary	240	5	Battleship	6
	2	Bill	225	14	Dog	NULL
	3	Jane	325	6	Car	7
	4	Norman	300	7	Thimble	3

Figure 3.7: Screenshot of Player Table after applying rule for ‘Community Chest 1’

Norman landed on ‘Community Chest 1’, therefore he gained £100 from winning the beauty contest. His location has also been updated.

Mary rolls a 6, then a 5;


```
Call LocationfromDiceRoll(NULL,'Mary',6);  
Call ActionifInJail(NULL,'Mary',6);
```




	message
▶	You are out of jail. Roll again!


Figure 3.8.1: Screenshot of message returned for player that was ‘In Jail’ and rolled a 6

```
Call SecondRollForJail(NULL,'Mary',5);  
  
Call Apply_Rules_Jail(NULL,'Mary',5);
```

100%  39:38

Result Grid 

Filter Rows:

Export: 


	message	new_position_name	old_p...
	The new Board Location for Mary is Oak House	Oak House	5

Figure 3.8.2: Screenshot of message returned after rolling the dice the second time

100%

22:50

Result Grid

Filter Rows:

Search

Edit:

Export/Import:

	PlayerID	PlayerName	BankBalance	CurrentLocationID	ChosenToken	BonusID	
▶	1	Mary	40	10	Battleship	6	
▶	2	Bill	225	14	Dog	NULL	
▶	3	Jane	325	6	Car	7	
▶	4	Norman	500	7	Thimble	3	

Figure 3.8.3: Screenshot of Player Table after applying rule for Property Location

```
SELECT * FROM Player;
```

After rolling a 6 and getting out of Jail, Mary rolled a 5 and landed on Oak house which is a property that not only has an owner(Norman), but the owner also owns another property of the same colour group as Oak House. This means Mary has to pay Norman double the rental cost (£200). **Figure 3.8.3** shows the transaction that has taken place.

Bill rolls a 6, then a 3;

100% 42:43

Result Grid Filter Rows: Search Export:

message	new_position_name	old_positi
The new Board Location for Bill is Uni Place . According to Rule R5 you have another chance, Roll Again	Uni Place	4

Figure 3.9.1: Screenshot of message returned after player rolled a 6

```
Call LocationfromDiceRoll(NULL,'Bill',6);
```

100% 1:45

Result Grid Filter Rows: Search Export:

message	new_position_name	old_position
The new Board Location for Bill is Community Chest 1	Community Chest 1	4

Figure 3.9.2: Screenshot of message returned after takes the second roll

```
Call SecondRoll(NULL,'Bill',3);
Call Apply_Rules_SecondRoll(NULL,'Bill',3);
```

100%

↕

22:56

Result Grid

Filter Rows:

Search

Edit:

Export/Import:

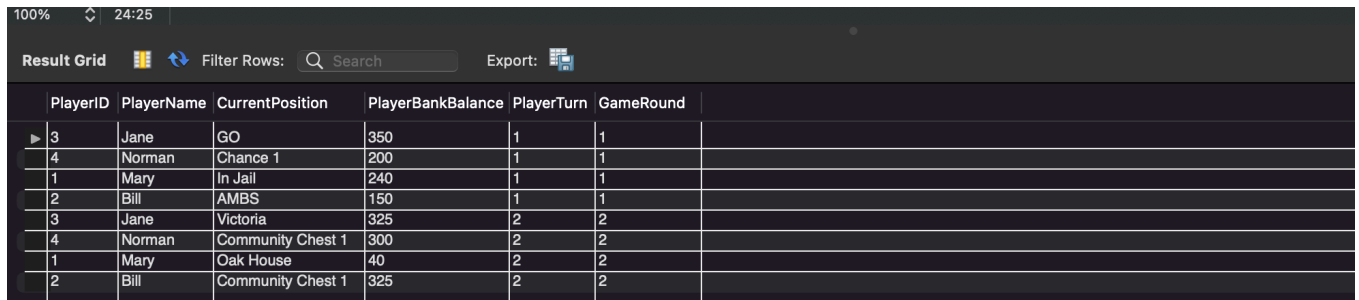
	PlayerID	PlayerName	BankBalance	CurrentLocationID	ChosenToken	BonusID	
▶	1	Mary	40	10	Battleship	6	
▶	2	Bill	325	7	Dog	3	
▶	3	Jane	325	6	Car	7	
▶	4	Norman	500	7	Thimble	3	

Figure 3.9.3: Screenshot of Player Table after applying rule for 'Community Chest 1'

SELECT * FROM Player;

After his second roll, Bill landed on 'Community Chest 1', therefore he gained £100 from winning the beauty contest. His location has been updated as well as the BonusID.

GameView after Round 2



	PlayerID	PlayerName	CurrentPosition	PlayerBankBalance	PlayerTurn	GameRound
▶	3	Jane	GO	350	1	1
■	4	Norman	Chance 1	200	1	1
■	1	Mary	In Jail	240	1	1
■	2	Bill	AMBS	150	1	1
■	3	Jane	Victoria	325	2	2
■	4	Norman	Community Chest 1	300	2	2
■	1	Mary	Oak House	40	2	2
■	2	Bill	Community Chest 1	325	2	2

Figure 4.0: Screenshot of GameView View after GameRound 2

END