Part 2 – LL(1) Parser Case 2

1. Exp → Exp | | Term | Term
2. Term → Term && Factor | Factor
3. Factor → Factor Compop Operand | Operand
4. Compop → > | = | <
5. Operand → ! Operand | id

There is left recursion in the given grammar highlighted above thus after removing left recursion the grammar will be:

1. Exp → Term Exp-dash
2. Exp-dash → | | Term Exp-dash | Ɛ
3. Term → Factor Term-dash
4. Term-dash → && Factor Term-dash | Ɛ
5. Factor → Operand Factor-dash
6. Factor-dash → Compop Operand | Ɛ
7. Compop → > | = | <
8. Operand → ! Operand | id

First and Follow Table

	First	Follow
Exp	{ ! , id }	{ $ }
Exp-dash	{ | | , Ɛ }	{ $ }
Term	{ ! , id}	{ | | , follow(Exp), follow(Exp-dash) } = { | | , $ }
Term-dash	{ && , Ɛ }	{ follow(Term) }} = { | | , $ }
Factor	{ ! , id }	{ && , follow(Term) }} = { && , | | , $ }
Factor-dash	{ > , = , < , Ɛ }	{ follow(Factor) }} = { && , | | , $ }
Compop	{ > , = , < }	{ ! , id }
Operand	{ ! , id }	{ follow(Factor-dash) , First (Factor-dash) } = { && , | | , $, > , < , = , Follow(Factor)} = {&& , | | , $, > , < , = }

Parse Table

	!	&&	| |	<	=	>	id	$
Exp	Exp → Term Exp-dash						Exp → Term Exp-dash	
Exp-dash			Exp-dash → || Term Exp-dash					Exp-dash → Ɛ
Term	Term → Factor Term-dash						Term → Factor Term-dash	
Term-dash		Term-dash → && Factor Term-dash	Term-dash → Ɛ					Term-dash → Ɛ
Factor	Factor → Operand Factor-dash						Factor → Operand Factor-dash	
Factor-dash		Factor-dash → Ɛ	Factor-dash → Ɛ	Factor-dash → Compop Operand	Factor-dash → Compop Operand	Factor-dash → Compop Operand		Factor-dash → Ɛ
Compop				Compop → <	Compop → =	Compop → >		
Operand	Operand → ! Operand						Operand → id	