Title: DB Assignment 6
Your Name: Kenny Chau
Date: 12/10/24

Query Type	Description	Dataset Size	Index Type	(Microseconds)
Point Query 1	SELECT count(*) FROM accounts WHERE branch_name = "Downtown"	50,000 Records	Without Indexes	37568.2000
	SELECT count(*) FROM accounts WHERE branch_name = "Downtown"		With Index (bname)	25384.2000
Point Query 2	SELECT count(*) FROM accounts WHERE balance = 10000	100,000 Records	Without Indexes	64833.8000
	SELECT count(*) FROM accounts WHERE balance = 10000		With Index (balance)	31228.7000
Point Query 3	SELECT count(*) FROM accounts WHERE account_type = "Savings"	150,000 Records	Without Indexes	57684.6000
	SELECT count(*) FROM accounts WHERE account_type = "Savings"		With Index (account_type)	57964.6000
Ranger Query 1	SELECT count(*) FROM accounts	50,000 Records	Without Indexes	47385.4000

	WHERE branch_name = "Downtown" AND balance BETWEEN 10000 AND 5000			
	SELECT count(*) FROM accounts WHERE branch_name = "Downtown" AND balance BETWEEN 10000 AND 5000		With Index (bname)	35933.7000
Range Query 2	SELECT count(*) FROM accounts WHERE balance BETWEEN 10000 AND 5000	100,000 Records	Without Indexes	68891.6000
	SELECT count(*) FROM accounts WHERE balance BETWEEN 10000 AND 5000		With Index (balance)	48216.000
Range Query 3	SELECT count(*) FROM accounts WHERE account_type = "Savings" and BALANCE BETWEEN 10000 AND 5000	150,000 Records	Without Indexes	64127.9000
	SELECT count(*) FROM accounts WHERE		With Index (account_type)	57661.6000

	account_type = "Savings" and BALANCE BETWEEN 10000 AND 5000			

Procedure 1: generate_accounts()

Purpose: The purpose of this procedure is to generate random accounts up to the number specified inside the procedure. The accounts contain a 6 number account number so that it can include numbers up to 999,999, without having to change the procedure from 5 numbers to 6. The accounts are generated by looping through a specified time and selecting an id, random branch name, account type, and balance. Afterwards the account is inserted into the accounts table.

Procedure 2: generate_execution_time()

Purpose: The purpose of this procedure is to time how long executing a sql query takes to execute. It starts a timer and begins an example query. After the query is finished, the time is recorded and displayed to the user.

Procedure 3: generate_report()

Purpose: The purpose of this procedure is to execute a given sql query ten times and record the average time it takes to execute a sql query. It goes through a loop that has a start and end time. The timer starts and executes the query. After ending the timer, the time is added to a sum of time taken. It divides the sum by 10 and returns the average back to the user.

Analysis: For a majority of the experiments, the results are expected as tables that had indexes had faster execution times than those that did not have indexes. Range queries typically were typically slower than point queries as point queries fetch specific rows, while range queries are expected to look for several rows that fit the criteria. One strange behavior that occurred is that when it approached 150,000 records, there was barely a difference in time between those with and without indexing. The possible reason for this is that there are so many rows of data that need to be looked at that the causes the fetching of data to slow down.