Mini Project Synopsis

HOTEL MANAGEMENT SYSTEM

Submitted as a part of course curriculum for

CORE COURSE IN

DATABASE MANAGEMENT SYSTEM

Under the guidance of
Prof Nivedita Kasturi

Submitted by -:
1) B Pravena – PES2UG19CS076
2) Bharath Kumar S P-PES2UG19CS087
3) Bhuvantej R – PES2UG19CS092

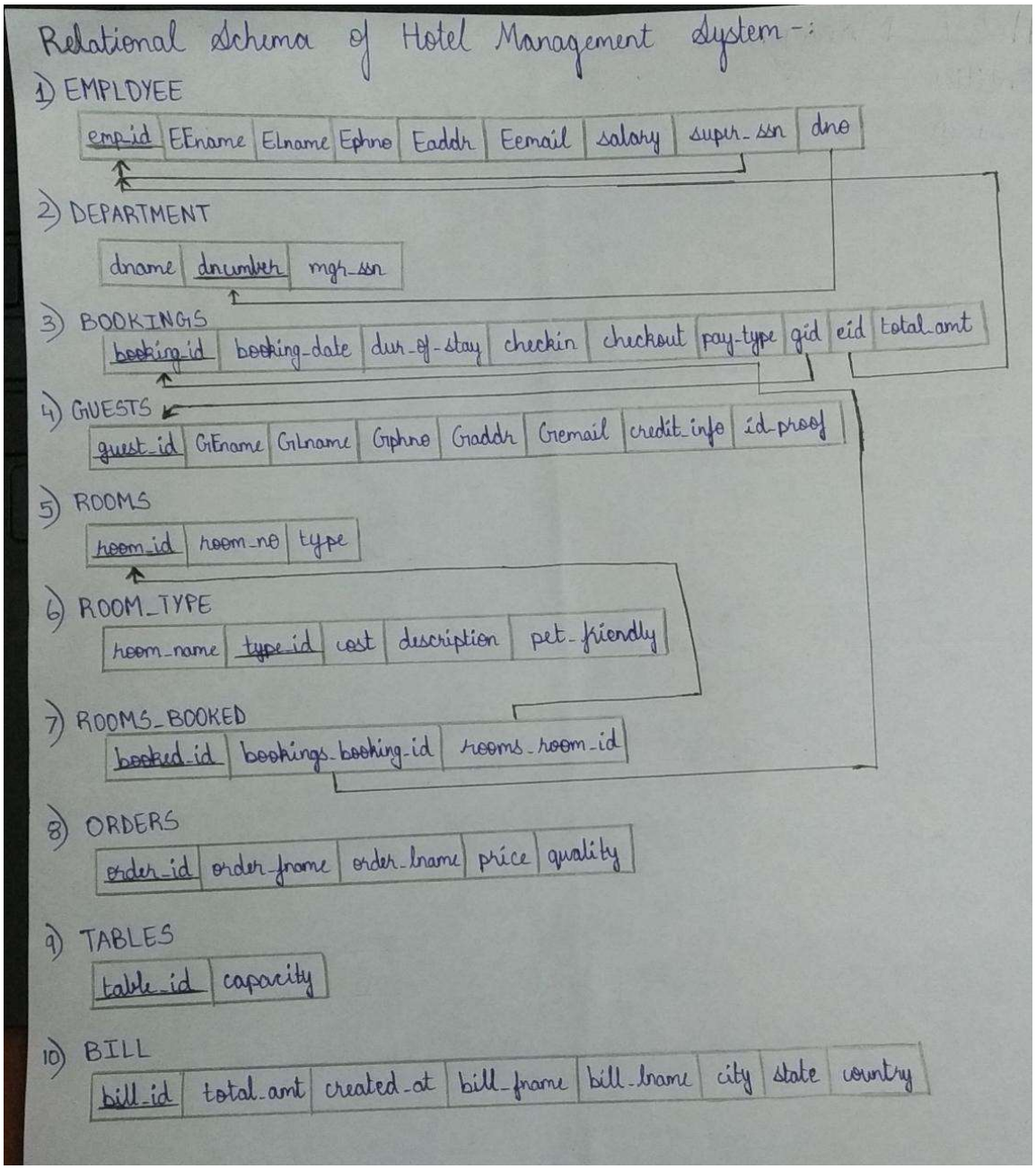
Department of Computer Science and Engineering

Pes University

Problem Statement -:

To create an efficient Hotel (lodge + restaurant) management system. The main objective is to manage the details of employees, bookings, guests at the lodge and customers at the restaurant, rooms, order, bill, etc. A customer can make reservations, change, or cancel reservations through the hotel website. When a customer makes reservations, based on availability employee allots room.

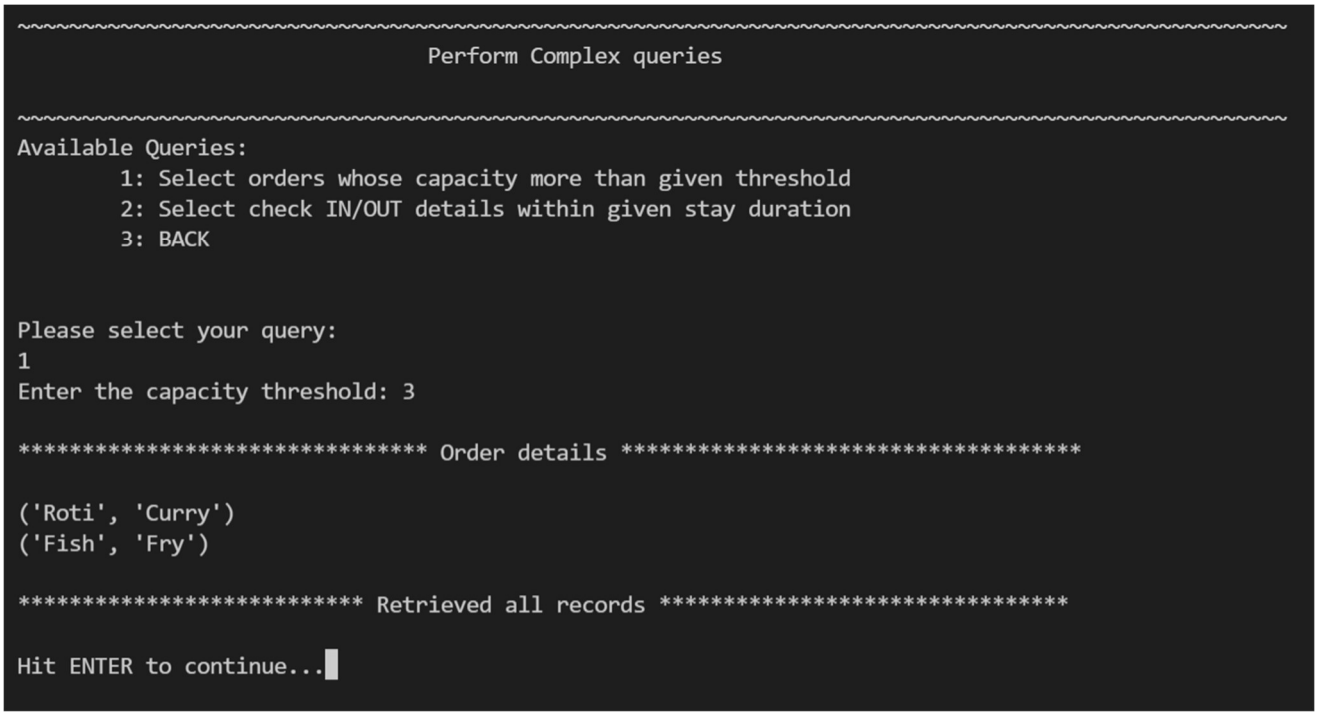
Relational Schema -:



Dependencies installed for the database connectivity -:

We have installed psycopg2. It is the most popular PostgreSQL database adapter for the Python programming language. It uses port number 5432. Its main features are the complete implementation of the Python DB API 2.0 specification and the thread safety (several threads can share the same connection). It was designed for heavily multi-threaded applications that create and destroy lots of cursors and make a large number of concurrent "INSERT"s or "UPDATE"s. It features client-side and server-side cursors, asynchronous communication and notifications, "COPY TO/COPY FROM" support. Psycopg 2 is both Unicode and Python 3 friendly.

Screenshots for the statements executed from the front end -:



```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                                Perform Complex queries

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Available Queries:
        1: Select orders whose capacity more than given threshold
        2: Select check IN/OUT details within given stay duration
        3: BACK


Please select your query:
2
Enter duration of stay: 4

****************** Check IN/OUT within duration *********************

(datetime.date(2021, 3, 22), datetime.date(2021, 3, 24))
(datetime.date(2021, 6, 11), datetime.date(2021, 6, 14))
(datetime.date(2021, 8, 19), datetime.date(2021, 8, 20))

******************* Retrieved all record******************

Hit ENTER to continue...
```

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
                                Perform Nested queries

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Available Queries:
        1: List employee under manager (using mgr_ssn)
        2: List guest using VISA
        3: Get the name and ID of employee having minimum salary.
        4: Employee having salary more than average salary of all departments.
        5: BACK


Please select your query:
1
Enter manager SSN to find employee under him 1

********************** Employee having Minimum Salary *******************************

('James', 'Borg')
('Ramesh', 'Narayan')
('Jonny', 'English')

****************************** Retrieved all records *********************************

Hit ENTER to continue...
```

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
                                Perform Nested queries

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Available Queries:
        1: List employee under manager (using mgr_ssn)
        2: List guest using VISA
        3: Get the name and ID of employee having minimum salary.
        4: Employee having salary more than average salary of all departments.
        5: BACK


Please select your query:
3

********************** Employee having Minimum Salary *******************************

(3, 'Franklin Wong', 30000.0)

****************************** Retrieved all records *********************************

Hit ENTER to continue...
```

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
                                Perform Nested queries

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Available Queries:
        1: List employee under manager (using mgr_ssn)
        2: List guest using VISA
        3: Get the name and ID of employee having minimum salary.
        4: Employee having salary more than average salary of all departments.
        5: BACK


Please select your query:
4

***************** Employee having Salary more than average salary of all department ****************************

(4, 'Alicia Zelaya', 65000.0)
(6, 'Ramesh Narayan', 60000.0)
(7, 'Jonny English', 60000.0)

****************************************** Retrieved all records ***********************************************

Hit ENTER to continue...
```

```
-----------------------------------------------------------------------------------------------
                                Perform Simple queries

-----------------------------------------------------------------------------------------------
Available Queries:
        1: List all Employees in our Hotel
        2: Select all employees whose salary is more than threshold
        3: Select all rooms types available (based on pet friendliness)
        4: List all Guests in our Hotel
        5: List room count with room_type
        6: BACK


Please select your query:
2
Enter salary threshold to search, 40000

****************** Employee Record ********************

(1, 'James', 'Borg', '888665555', '450 Stone, Houston,TX', 'james.borg@gmail.com', 55000.0, None, 1)
(4, 'Alicia', 'Zelaya', '988455255', '3321 Castle,Spring,TX', 'alica.zelaya@yaahoo.com', 65000.0, None, 2)
(5, 'Jennifer', 'Wallace', '9867585821', '291 Berry, Bellaire,TX', 'jennifer@gmail.com', 45000.0, 2, 2)
(6, 'Ramesh', 'Narayan', '8884455920', '975 Fire Oak, Humble, TX', 'ramesh.narayan@gmail.com', 60000.0, 1, 1)
(7, 'Jonny', 'English', '998665533', '5631 Rice,Houston,TX', 'johnny.english@gmail.com', 60000.0, 1, 1)

*************** Retrieved all record*****************

Hit ENTER to continue...
```

```
-----------------------------------------------------------------------------------------------
                                Perform Simple queries

-----------------------------------------------------------------------------------------------
Available Queries:
        1: List all Employees in our Hotel
        2: Select all employees whose salary is more than threshold
        3: Select all rooms types available (based on pet friendliness)
        4: List all Guests in our Hotel
        5: List room count with room_type
        6: BACK


Please select your query:
3
Are u looking for pet friendly room? (y/n) n

****************** Rooms type List ********************

(21, 'Single', 0)
(22, 'Double', 0)
(25, 'Single+Balcony', 0)
(27, 'Double+Balcony', 0)

*************** Retrieved all record*****************

Hit ENTER to continue...
Loading.....
```

```
                              Perform Simple queries

-------------------------------------------------------------------------------------------------
Available Queries:
        1: List all Employees in our Hotel
        2: Select all employees whose salary is more than threshold
        3: Select all rooms types available (based on pet friendliness)
        4: List all Guests in our Hotel
        5: List room count with room_type
        6: BACK


Please select your query:
4

******************************** Guest in Hotel ***********************************

(1, 'Salman', 'Khan', '9900012345', '112,mumbai', 'sallubhai@gmail.com', 'visa', 'voter id card')
(2, 'Mithali', 'Raj', '9900054321', '201,bangalore', 'mithaliraj@gmail.com', 'rupay', 'pan card')
(3, 'Smrithi', 'Mandhana', '9945724378', '420,pune', 'mandhana@gmail.com', 'maestro', 'aadhar card')
(4, 'MS', 'Dhoni', '8892736433', '007,jharkhand', 'Helicopter_six@gmail.com', 'maestro', 'aadhar card')
(5, 'Gautham', 'Gambhir', '9743296116', '70,Delhi', 'gauthi07@gmail.com', 'rupay', 'pan card')
(6, 'Rohit', 'Sharma', '9900011111', '264,Hyderabad', 'doublecentury@gmail.com', 'visa', 'voter id card')
(7, 'Virat', 'Kholi', '9000000143', '100,Delhi north', 'viratkohli@yahoo.com', 'visa', 'aadhar card')
(8, 'Sachin', 'Tendulkar', '9988776655', '10,Kerala', 'godofcricket@gmail.com', 'rupay', 'aadhar card')

***************************** Retrieved all records ********************************

Hit ENTER to continue...
```

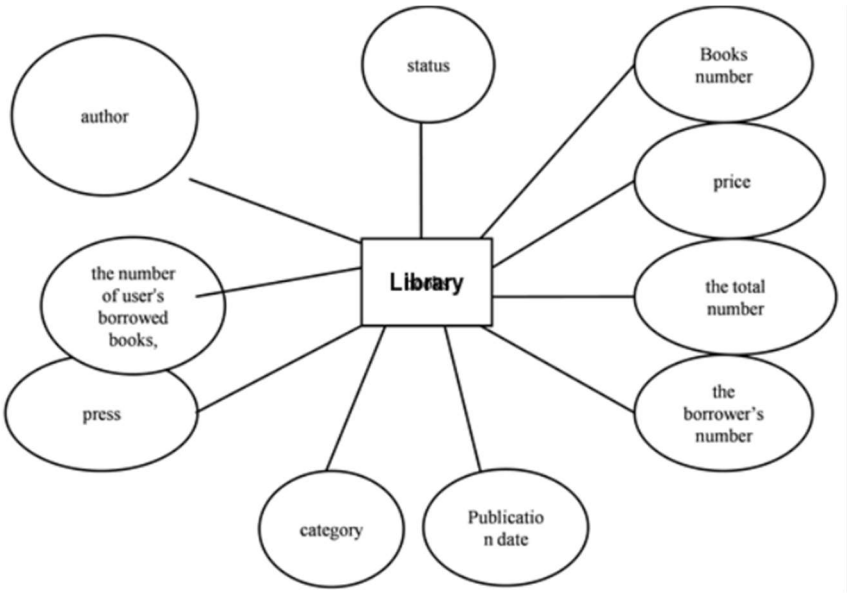
```
*************************************************************************************************
                              Welcome to our Hotel DataBase

*************************************************************************************************
Available Queries:
        1: Complex query menu
        2: Nested query menu
        3: Simple query menu
        4: Query by DBA
        5: QUIT


Please select your query:
4

Please enter the DBA pass code: *******
Enter your query: SELECT * FROM BOOKINGS;

******************************** Query Output ***********************************

(2, datetime.date(2021, 1, 25), 8, datetime.date(2021, 2, 2), datetime.date(2021, 2, 10), 'Cash', 2, 2, 2000.5)
(3, datetime.date(2021, 3, 10), 2, datetime.date(2021, 3, 22), datetime.date(2021, 3, 24), 'paytm', 3, 3, 2300.0)
(4, datetime.date(2021, 4, 12), 11, datetime.date(2021, 4, 18), datetime.date(2021, 4, 29), 'Card', 4, 4, 1500.0)
(5, datetime.date(2021, 5, 18), 5, datetime.date(2021, 5, 20), datetime.date(2021, 5, 25), 'Cash', 5, 5, 1500.0)
(6, datetime.date(2021, 6, 12), 3, datetime.date(2021, 6, 11), datetime.date(2021, 6, 14), 'Cash', 6, 6, 1500.0)
(7, datetime.date(2021, 6, 30), 9, datetime.date(2021, 7, 5), datetime.date(2021, 7, 14), 'paytm', 7, 7, 1500.0)
(8, datetime.date(2021, 7, 27), 1, datetime.date(2021, 8, 19), datetime.date(2021, 8, 20), 'Card', 8, 8, 1500.0)

***************************** Retrieved all records ********************************
```

```
**********************************************************************************************************
                                Welcome to our Hotel DataBase

**********************************************************************************************************
Available Queries:
        1: Complex query menu
        2: Nested query menu
        3: Simple query menu
        4: Query by DBA
        5: QUIT


Please select your query:
5

Thanks for visiting.

PS D:\DBMS_Assignment\Assignment_4>
```

Changes in Business or Application or expansion that might lead to -:

- **Schema Changes –** If the business has a really good revenue, then, to provide better facilities in the Hotel we can include a library, children's park, swimming centre, spa, valet/ car parking etc each of which will have its own tables in the database. We can also open many branches which in change will require its own set of tables.

Example for library



- **Constraint Changes –** In case of expansion we can add extra security levels which we can apply by adding constraints such as matching guest real-time face and uploaded selfie photo on database. We can add other constraints such as guest can receive key only after checking in at front desk, swimming centre will be open only if customer has not yet checked out and time is between 10AM and 10PM, borrow a book from library only if previous books are returned and dues paid.

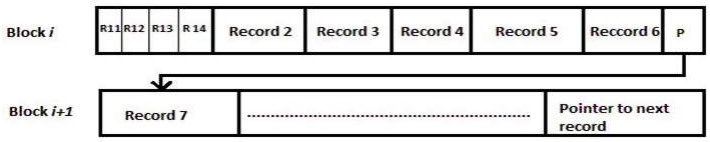
- **DBMS migration –** In case of expansion especially if we have to handle more branches, we can switch to columnar form of database which is available on PostgreSQL itself. This helps us to retrieve and update data instantaneously. (More about it in next page)

With the existing design of your database, if you have to migrate to any No-SQL variety, then which one will be your choice? Why?

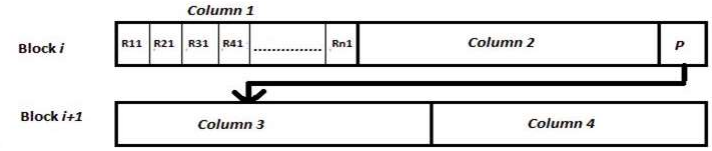
If we have to convert to a NoSQL variety, then our choice would be **Column-Oriented Database.** This is because Postgres is a full featured open-source DB that has both traditional row-based storage (sometimes called "heap files") as well as a columnar store extension. This will help us easily migrate to columnar format without having to use another form of database. A relational database is ideal for transactional applications because it stores rows of data whereas a columnar database is preferred for analytical applications because it allows for faster retrieval of columns of data. Columnar databases are designed for data warehousing and big data processing because they scale using distributed clusters of low-cost hardware to increase throughput. This is especially useful if the hotel has started having a large number of customers or, if they opened many branches or, if we want to conduct an analysis about surge in customers, how much to expect during holiday season, etc.

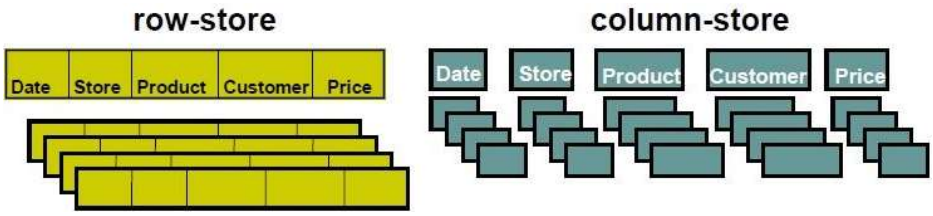
RDBMS vs. Columnar Oriented DBMS (Physical Level)

Row oriented



Column oriented





We in specific would have chosen either **PostgreSQL's columnar database** or **Apache Cassandra**.

The features and advantages of **PostgreSQL's columnar database-:**

- Compression: Reduces in-memory and on-disk data size by 2-4 times. Can be extended to support different codecs.
- Column projections: Only reads column data relevant to the query. Improves performance for I/O bound queries.
- PostgreSQL's query optimizer uses these stats to evaluate different query plans and pick the best one.
- The columnar store extension is self-contained. If you're a PostgreSQL user, you can get the entire source code and build using the instructions on their GitHub page. We can even join columnar store and regular Postgres tables in the same SQL query.

The main features and advantages of **Apache Cassandra** are -:

- It is scalable, consistent and fault tolerant.
- It is key-value as well as column-oriented database.
- Elastic scalability: Cassandra allows adding more hardware to accommodate more customers and more data as per requirement.
- Cassandra is continuously available for critical business applications that cannot afford single point of failure.
- Fast linear-scale performance: Cassandra increases throughput as the number of nodes in the cluster is increased. Therefore, it provides a quick response time.
- Flexible data storage: Cassandra handles all possible data formats including: structured, semi-structured, and unstructured. It can dynamically provide changes to data structures according to user need.
- Easy data distribution: Cassandra provides the flexibility to distribute data where user need by replicating data across multiple data centres.
- Transaction support: Cassandra supports properties like Atomicity, Consistency, Isolation, and Durability (ACID).
- Fast writes: Cassandra was designed to run on cheap commodity hardware. It performs fast writes and can store hundreds of terabytes of data, without sacrificing the read efficiency

Contributions -:

1) **B. Pravena** – PES2UG19CS076 – user input for complex queries, new nested query code, testing the final output, compilation of report (4 days)
2) **Bharath Kumar S P -** PES2UG19CS087 – worked on password hiding in user interface, user input for simple queries, DB to Front end connectivity (5 days)
3) **Bhuvantej R** – PES2UG19CS092 – user input for nested query, reusable front-end code for various functions (4 days)