Task

1. Probable use cases of joins in real time applications (it can be in the broad perspective) that where give a real time practical applications in database.

JOIN operations can also be applied in college hostel management systems.

Room allocation

Used to match student’s preferences like roommates or specific room types, with available rooms in the hostel.

Fee payment

Join can link student profiles, fee payment record and financial transactions.

Maintenance

Maintenance requests and work orders.

For example:

Inner Join: An Inner Join can be used to match student profiles with the room allocation data, visitor logs with resident information.

Left Join: A left Join can be used to retrieve all student profiles and their perspective room allocations.

Right Join: A Right Join can be used to retrieve all room allocations and their perspective student profiles.

Full Outer Join: Lists of all student profiles and their room allocations, including students without a room and unallocated rooms.

Cross Join: All possible combinations of students and room types.

|  |  |  |
| --- | --- | --- |
| ID | Name | RoomID |
| 1 | Kamalendu | 101 |
| 2 | Sufaila | 102 |
| 3 | Maneesha | 103 |

Student table

Room table

|  |  |
| --- | --- |
| ID | RoomNo |
| 101 | A101 |
| 102 | B202 |
| 103 | C303 |

HostelFee table

|  |  |
| --- | --- |
| StudentID | Amount |
| 1 | 500 |
| 3 | 300 |

InnerJoin

This inner join combines the “student” and “Room” tables based on the matching RoomID and ID columns. It retrieves the names of students along with their assigned room numbers.

|  |  |
| --- | --- |
| Name | RoomNo |
| Kamalendu | A101 |
| Sufaila | B202 |
| Maneesha | C303 |

LeftJoin

This left join retrieves the names of all students from the “Student” table and their corresponding hostel fee amounts from the “HostelFee” table. It includes all students even if they do not have a matching record in the “HostelFee” table

|  |  |
| --- | --- |
| Name | Amount |
| Kamalendu | 500 |
| sufaila | Null |
| Maneesha | 300 |

RightJoin

This right join retrieves the names of students and their corresponding hostel fee amounts from the “HostelFee” table. It includes all records from the “Hostelfee” table even if there is no matching student record in the “Student” table.

|  |  |
| --- | --- |
| Name | Amount |
| Kamalendu | 500 |
| Maneesha | 300 |

FullOuterJoin

This full outer join retrieves the names of students and their corresponding hostel fee amounts. It includes all student records and matches them with the respective hostel fee records.

|  |  |
| --- | --- |
| Name | Amount |
| Kamalendu | 500 |
| Sufaila | Null |
| Maneesha | 300 |

1. Give a read on normalization and types of normalization.

First Normal Form (1NF)

A relational entity satisfies the requirement of 1NF if every instance of the entity contains only one value, but never multiple repeating attributes. It eliminates repeating groups and each column has a unique name.

|  |  |  |
| --- | --- | --- |
| Student Id | Name | Subjects |
| 1 | Kamalendu | Maths, English, History |
| 2 | Maneesha | Science, Maths |

|  |  |
| --- | --- |
| Student Id | Name |
| 1 | Kamalendu |
| 2 | Maneesha |

Students table

Students table

Students\_Subjects table

|  |  |
| --- | --- |
| Student Id | Subjects |
| 1 | Maths |
| 1 | English |
| 1 | History |
| 2 | Science |
| 2 | Maths |

Second Normal Form (2 NF)

It must be in 1NF and all non-key attributes fully dependent on primary key.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Customer Id | Product Name | Product Id | Order Id | Quantity |
| 101 | A | 1 | 1 | 3 |
| 101 | B | 2 | 1 | 2 |
| 102 | A | 1 | 2 | 5 |

Orders table

Product name depend on product Id so split the table.

Orders table

|  |  |  |  |
| --- | --- | --- | --- |
| Order Id | Customer Id | Product Id | Quantity |
| 1 | 101 | 1 | 3 |
| 1 | 101 | 2 | 2 |
| 2 | 102 | 1 | 5 |

Products

|  |  |
| --- | --- |
| Product Name | Product Id |
| A | 1 |
| B | 2 |

Third Normal Form (3NF)

* A relation will be in 3NF if it is in 2NF and not contain any transitive partial dependency.
* 3NF is used to reduce the data duplication.
* If there is no transitive dependency for non-prime attributes, then the relation must be in third normal form.

A relation is in third normal form if it holds at least one of the following conditions for every non-trivial function dependency X → Y.

1. X is a super key.
2. Y is a prime attribute, i.e., each element of Y is part of some candidate key.

|  |  |  |  |
| --- | --- | --- | --- |
| Employee Id | Employee Name | Department Id | Department Name |
| 1 | Kamalendu | 1 | Sales |
| 2 | Maneesha | 2 | Marketing |

Super key: {Employee Id}, {Employee Id, Employee Name}, { Employee Id, Employee Name ,Department Id}, { Employee Id, Employee Name, Department Id, Department Name }

A transitive dependency between the Department Name and the Department Id. The Department Name depends on the Department Id, but it is not directly dependent on the Employee Id. So split the table.

|  |  |  |
| --- | --- | --- |
| Employee Id | Employee Name | Department Id |
| 1 | Kamalendu | 1 |
| 2 | Maneesha | 2 |

|  |  |
| --- | --- |
| Department Id | Department Name |
| 1 | Sales |
| 2 | Marketing |