**Phase 2 Report**

**Instacart Market Basket Analysis**

**Group 36**

* **Aisles**Graphical user interface, application, Teams

  Description automatically generated
* Document-oriented model -

{

“aisle\_id”: \_,

“aisle”:\_

}

* Relational model-  
   Aisle(aisle\_id, aisle\_name)
* **Departments**Graphical user interface, application

  Description automatically generated
* Document-oriented model -

{

“department\_id”:\_,

“department:\_

}

* Relational model-

Department(department\_id, department\_name)

* **Order**Graphical user interface, application, Teams

  Description automatically generated
* Document-oriented model -

{

“order\_id”:\_,

“user\_id:”\_,

“order\_number”:\_,

“order\_dow”:\_,

“order\_hour\_of\_day”:\_,

“days\_since\_prior\_order”:\_

}

* Relational model-

Order(order\_id, user\_id, order\_number, order\_dow, order\_hour\_of\_day, days\_since\_prior\_order)

* **Order\_product**Graphical user interface, application

  Description automatically generated
* Document-oriented model -

{

“order\_id”:\_,

“product\_id:\_,

“add\_to\_cart\_order”:\_,

“reordered”:\_

}

* Relational model-

Order\_Product(order\_id, product\_id, add\_to\_cart\_order, reordered)

* **Product**Graphical user interface, text, application

  Description automatically generated
* Document-oriented model -

{

“product\_id”:\_,

“product\_name”:\_,

“aisle\_id”:\_,

“department\_id”:\_

}

* Relational model-

Product(product\_id, product\_name, aisle\_id, department\_id)

* **Users**Graphical user interface, application

  Description automatically generated
* Document-oriented model -

{

“user\_id”:\_,

“name”:\_

}

* Relational model-

User(user\_id, name)

**Queries –**

Our tables:

Text

Description automatically generated

1. Find the top 10 products that were most frequently reordered by customers:

Shape

Description automatically generated with low confidence

2)Find the average days since a customer's last order for each department:

Text

Description automatically generated

3)Find the top 10 aisles with the highest percentage of products that were reordered:

Text

Description automatically generated with medium confidence

4)Find the number of products in each department that were purchased during each hour of the day:

Text

Description automatically generated

1. Find the top 10 most popular product pairs that were purchased together by customers:

Text

Description automatically generated

Time taken by these 5 queries before adding indexing:

Graphical user interface, text

Description automatically generated

Time taken by the same queries after adding indexing:

Graphical user interface, text, application

Description automatically generated

As we can see the time got decreased by a considerable amount for query1, and 3. For the last query it still takes a lot of time, as it deals with 4 tables.

**Functional Dependances –**

In the given schema, some of the functional dependencies are as follows:

* User: user\_id -> name
* Order: order\_id -> user\_id, order\_number, order\_dow, order\_hour\_of\_day, days\_since\_prior\_order
* Product: product\_id -> product\_name, aisle\_id, department\_id
* Aisle: aisle\_id -> aisle\_name
* Department: department\_id -> department\_name
* Order\_Product: (order\_id, product\_id) -> add\_to\_cart\_order, reordered

**Normalization -**

In the given schema, the tables are already in 1NF, as each cell has a single value, and each column has a unique name. However, the tables may not be in 2NF or 3NF, as some attributes may be functionally dependent on non-primary key columns.

For example, in the Order table, the days\_since\_prior\_order attribute is functionally dependent on the order\_id, but not on the user\_id. This violates the requirements of 2NF.

To normalize the tables, the redundant attributes should be moved to separate tables, and relationships between tables should be established through foreign keys.

For example, in the Order table, the days\_since\_prior\_order attribute should be moved to a separate table, which is related to the Order table through the order\_id primary key.

Similarly, in the Order\_Product table, the add\_to\_cart\_order attribute is functionally dependent only on the product\_id and not on the order\_id, so it should be moved to a separate table related to the Product table through the product\_id primary key.

By normalizing the tables, the database becomes more efficient and easier to maintain, as it eliminates redundant data and ensures data integrity.