Database Management
MySQL Report On
Restaurant Billing System

1. *Rationale:-*

- This software can be used by a wide variety of outlets (Retailers and Wholesaler's) to automate the process of manually maintaining the records related to the subject.

2. *Aim of the Project:-*

- We are creating database of Restaurant bill in a Restaurant for keeping records of different items as well as of customers also.

❖ ***Benefits of the Project:-***

- Generates financial statements fast and accurately.
- Only authorized users have access to the database and hence the data is secured.
- Similar type of database can be used by different organizations.

3. *Course Outcomes Addressed:-*

- Create Database using SQL Commands.
- Manage Database Using SQL Commands.
- Implement Advanced SQL Commands.

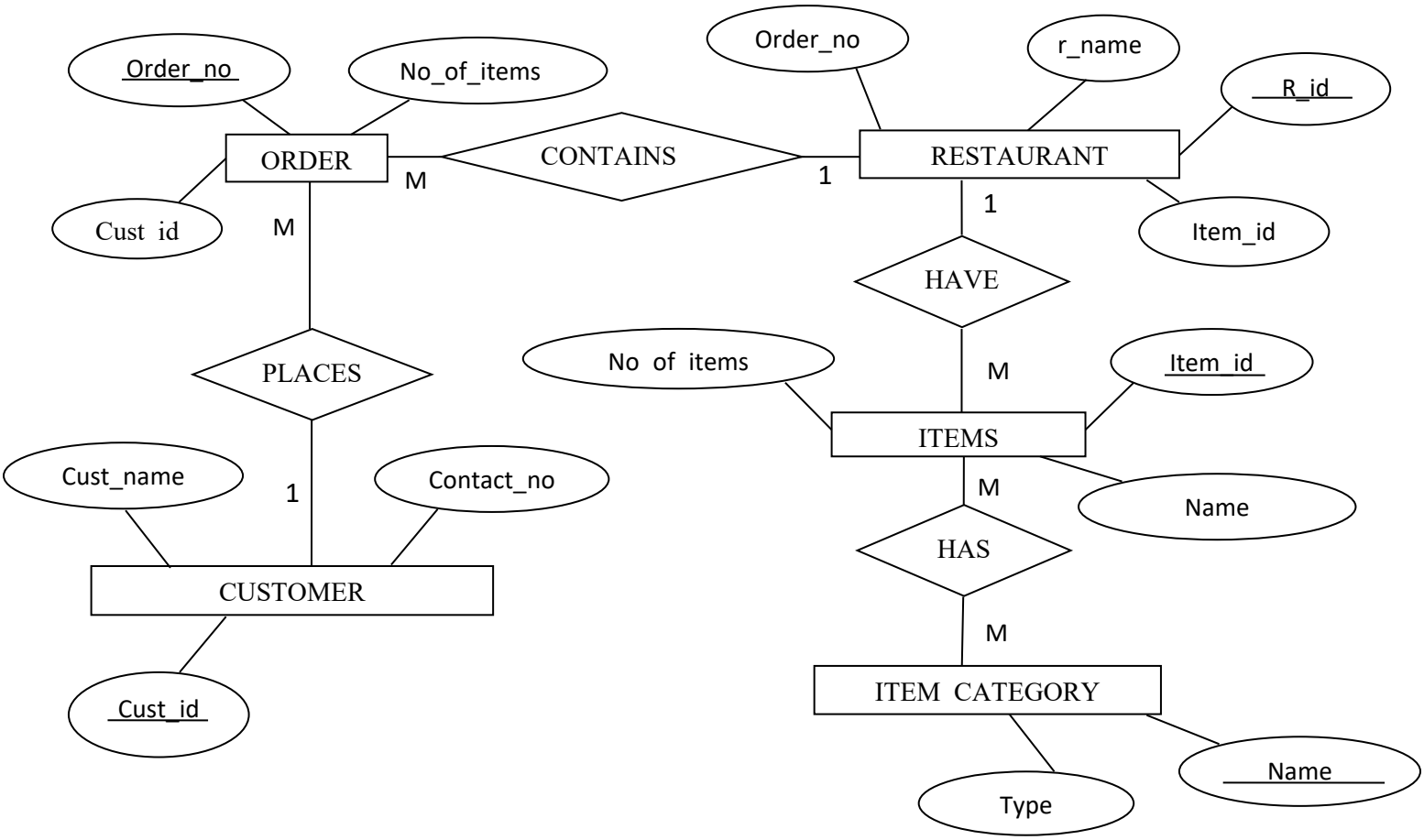
4. *Literature Review:-*

❖ ***Theory of database:-***

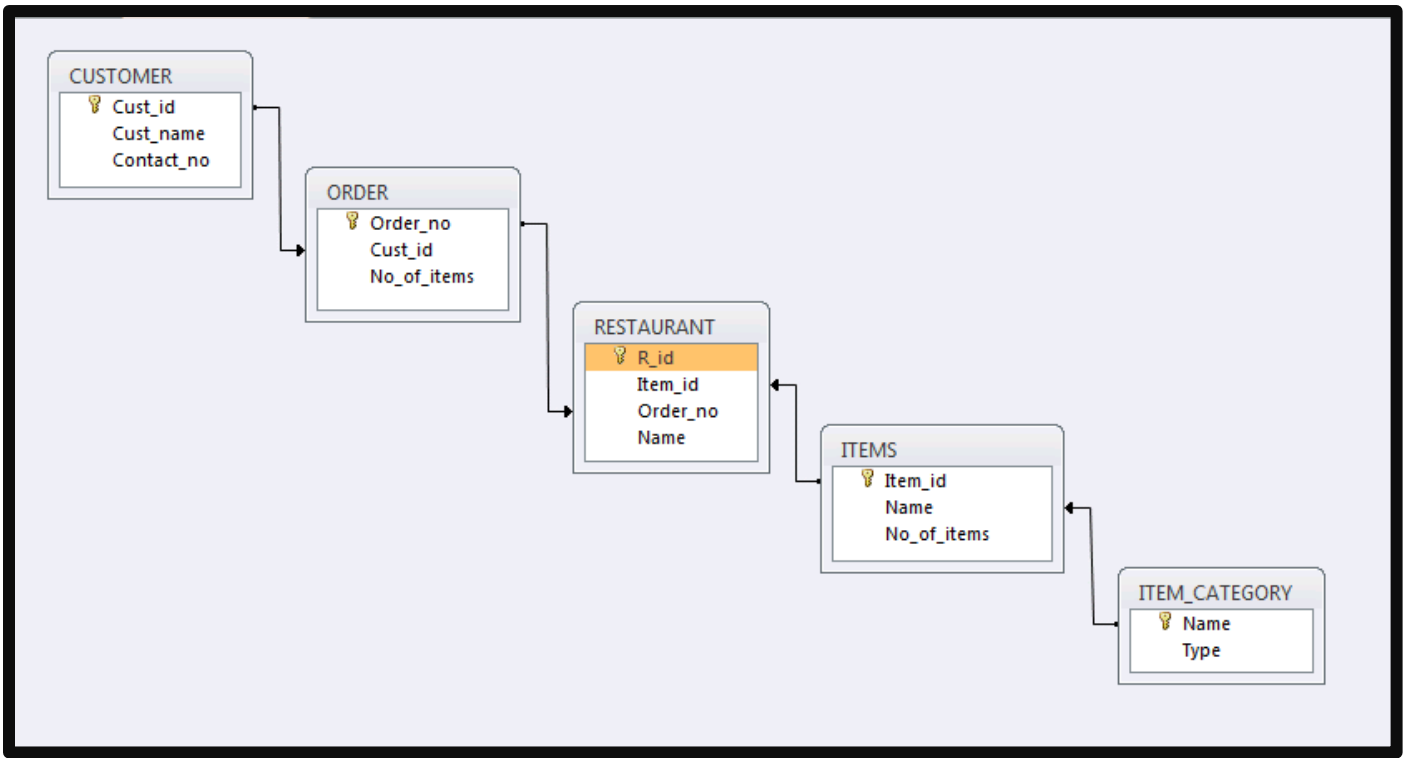
- This project is basically updating the manual restaurant billing system to automated, so that the organization can manage the record in efficient and organized form.
- The main goal of this project is to maintain the records of items.
- This is helpful them to maintain day to day transactions.
- This is helpful in generating financial statements fast and accurately.

5. *Actual Methodology Followed:-*

A. Creation of ER Diagram:-



A. Schema Diagram:-



B. Creation and Insertion of tables in My SQL:-

a) Table Customer:

```
SQL> create table customer
  2  ( cust_id numeric(10) primary key,
  3  cust_name varchar2(20),
  4  contact_no numeric(10)
  5  );

Table created.
```

```
SQL> insert into customer values
  2  (&cust_id,'&cust_name',&contact_no);
Enter value for cust_id: 101
Enter value for cust_name: Rufee
Enter value for contact_no: 7777722222
old   2: (&cust_id,'&cust_name',&contact_no)
new   2: (101,'Rufee',7777722222)

1 row created.
```

```
SQL> /
Enter value for cust_id: 102
Enter value for cust_name: Hardik
Enter value for contact_no: 9876543210
old   2: (&cust_id,'&cust_name',&contact_no)
new   2: (102,'Hardik',9876543210)

1 row created.
```

```
SQL> /
Enter value for cust_id: 103
Enter value for cust_name: Sohan
Enter value for contact_no: 9988776655
old   2: (&cust_id,'&cust_name',&contact_no)
new   2: (103,'Sohan',9988776655)

1 row created.
```

```
SQL> /
Enter value for cust_id: 104
Enter value for cust_name: Zuha
Enter value for contact_no: 8877665544
old   2: (&cust_id,'&cust_name',&contact_no)
new   2: (104,'Zuha',8877665544)

1 row created.
```

b) Table Order:

```
SQL>  Create table orderss
  2   (  order_no numeric(10) primary key,
  3   No_of_items numeric(10),
  4   cust_id numeric(10) constraint FKu references customer(cust_id) ON DELETE CASCADE
  5   );

Table created.
```

```
SQL> Insert into orderss values
  2   (&order_no,&no_of_items,&cust_id);
Enter value for order_no: 1
Enter value for no_of_items: 2
Enter value for cust_id: 101
old   2:  (&order_no,&no_of_items,&cust_id)
new   2:  (1,2,101)

1 row created.
```

```
SQL> /
Enter value for order_no: 2
Enter value for no_of_items: 3
Enter value for cust_id: 102
old   2:  (&order_no,&no_of_items,&cust_id)
new   2:  (2,3,102)

1 row created.
```

```
SQL> /
Enter value for order_no: 3
Enter value for no_of_items: 2
Enter value for cust_id: 103
old   2:  (&order_no,&no_of_items,&cust_id)
new   2:  (3,2,103)

1 row created.
```

```
SQL> /
Enter value for order_no: 4
Enter value for no_of_items: 6
Enter value for cust_id: 104
old   2:  (&order_no,&no_of_items,&cust_id)
new   2:  (4,6,104)

1 row created.
```

c) Table Restaurant:

```
SQL> create table restaurant
  2  ( r_id numeric(10) primary key,
  3  r_name varchar2(20),
  4  order_no numeric(10),
  5  item_id numeric(10)
  6  );

Table created.
```

```
SQL> insert into restaurant values
  2  (&r_id,'&r_name',&order_no,&item_id);
Enter value for r_id: 111
Enter value for r_name: Allstar
Enter value for order_no: 1
Enter value for item_id: 121
old   2: (&r_id,'&r_name',&order_no,&item_id)
new   2: (111,'Allstar',1,121)

1 row created.
```

```
SQL> /
Enter value for r_id: 222
Enter value for r_name: Sai Krupa
Enter value for order_no: 2
Enter value for item_id: 122
old   2: (&r_id,'&r_name',&order_no,&item_id)
new   2: (222,'Sai Krupa',2,122)

1 row created.
```

```
SQL> /
Enter value for r_id: 333
Enter value for r_name: Allstar
Enter value for order_no: 3
Enter value for item_id: 123
old   2: (&r_id,'&r_name',&order_no,&item_id)
new   2: (333,'Allstar',3,123)

1 row created.
```

```
SQL> /
Enter value for r_id: 444
Enter value for r_name: Allstar
Enter value for order_no: 4
Enter value for item_id: 124
old   2: (&r_id,'&r_name',&order_no,&item_id)
new   2: (444,'Allstar',4,124)

1 row created.
```

d) Table Items:

```
SQL> create table items
  2  ( item_id numeric(10) primary key,
  3  name varchar2(20),
  4  No_of_items numeric(10)
  5  );

Table created.
```

```
SQL> insert into items values
  2  (&item_id,'&name',&No_of_items);
Enter value for item_id: 121
Enter value for name: dry paneer chilli
Enter value for no_of_items: 2
old   2: (&item_id,'&name',&No_of_items)
new   2: (121,'dry paneer chilli',2)

1 row created.
```

```
SQL> /
Enter value for item_id: 124
Enter value for name: ZamZAm Pulav
Enter value for no_of_items: 6
old   2: (&item_id,'&name',&No_of_items)
new   2: (124,'ZamZAm Pulav',6)

1 row created.
```

```
SQL> /
Enter value for item_id: 122
Enter value for name: biryani
Enter value for no_of_items: 3
old   2: (&item_id,'&name',&No_of_items)
new   2: (122,'biryani',3)

1 row created.
```

```
SQL> /
Enter value for item_id: 123
Enter value for name: Murg Musallam
Enter value for no_of_items: 2
old   2: (&item_id,'&name',&No_of_items)
new   2: (123,'Murg Musallam',2)

1 row created.
```

e) Table Item_Category:

```
SQL> create table item_category
  2  ( name varchar2(20),
  3  type varchar2(20)
  4  );

Table created.
```

```
SQL>  insert into item_category values
  2   ('&name','&type');
Enter value for name: dry paneer chilli
Enter value for type: veg
old   2:  ('&name','&type')
new   2:  ('dry paneer chilli','veg')

1 row created.
```

```
SQL> /
Enter value for name: biryani
Enter value for type: non-veg
old   2:  ('&name','&type')
new   2:  ('biryani ','non-veg')

1 row created.
```

```
SQL> /
Enter value for name: Murg Musallam
Enter value for type: non-veg
old   2:  ('&name','&type')
new   2:  ('Murg Musallam','non-veg')

1 row created.
```

```
SQL> /
Enter value for name: ZamZam Pulav
Enter value for type: non-veg
old   2:  ('&name','&type')
new   2:  ('ZamZam Pulav','non-veg')

1 row created.
```

C. Performing SQL Commands like Alter, Select, Update, Delete, View and Cursor on Restaurant:

a) ALTER:-

- To modify the table Structure, we use alter command. After creating a table, one may need to change the table, so these SQL Alter commands make changes to the definition of an SQL table.

```
SQL> alter table restaurant
  2  modify rest_name varchar2(15);

Table altered.
```

```
SQL> alter table restaurant
  2  drop (address);

Table altered.
```

```
SQL> alter table restaurant
  2  rename column r_name to rest_name;

Table altered.
```

```
SQL> alter table restaurant
  2  add unique(order_no);

Table altered.
```

```
SQL> alter table restaurant
  2  drop (address);

Table altered.
```

b) ***SELECT:-***

- Select Command is used to display the entire table along with its data.

```
SQL> select * from restaurant;

      R_ID REST_NAME         ORDER_NO    ITEM_ID
---------- --------------- ---------- ----------
       111 Allstar                 1        121
       222 Sai Krupa               2        122
       333 Allstar                 3        123
       444 Allstar                 4        124
```

c) ***UPDATE:-***

- SQL Update command is required to make changes or modifications in the records of the table. The User can modify the existing data stored in the table.

```
SQL> update restaurant
  2  set rest_name='Allstar'
  3  where r_id=222;

1 row updated.
```

d) ***DELETE:-***

- SQL Delete command is used to remove the rows in a table. It removes one or more records, 1 by 1 from the table.

```
SQL> delete from restaurant
  2  where r_id=444;

1 row deleted.
```

e) ***VIEW:-***

- A view is a logical table based on a table or another view. It represents subsets of data from one or more tables.
- A view contains no data of its own but is like a window through which data from tables can be viewed or changed. The view is stored as a select statement in the data dictionary.

```
SQL> create view r_view
  2  as select * from restaurant;

View created.

SQL> select * from r_view;

      R_ID REST_NAME         ORDER_NO    ITEM_ID
---------- --------------- ---------- ----------
       111 Allstar                 1        121
       222 Allstar                 2        122
       333 Allstar                 3        123
```

f) ***CURSOR:-***

- Oracle creates a memory area known as context area for processing SQL statements which contains all information needed for processing the statement.

- A cursor is a pointer to this context area. PL/SQL controls the context area through a cursor. The Cursor holds the rows but can process one row at times which is returned by SQL Statement. The set of rows that the cursor holds is referred as Active Dataset.
- Cursor is of following types:

a. Implicit Cursor.
b. Explicit Cursor.

- Following is an example of Explicit Cursor which is executed in My SQL:-

```
SQL>  declare
  2   rid restaurant.r_id%type;
  3   cursor rest_cursor is select r_id  from restaurant;
  4   begin
  5   open rest_cursor;
  6   loop
  7   fetch rest_cursor into rid;
  8    exit when rest_cursor%notfound;
  9    end loop;
 10    dbms_output.put_line(rid);
 11    close rest_cursor;
 12    end;
 13  /
333

PL/SQL procedure successfully completed.
```

D. <u>Performing Commands like Grant and Revoke in SQL:-</u>

a. <u>Grant:-</u>

- SQL GRANT is a command used to provide access or privileges on the database objects to the users.
- The Syntax for the GRANT command is:
  GRANT privilege_name
  ON object_name
  TO {user_name |PUBLIC |role_name}
  [WITH GRANT OPTION];

b. <u>Revoke:-</u>

- The REVOKE command removes user access rights or privileges to the database objects.
- The Syntax for the REVOKE command is:
  REVOKE privilege_name
  ON object_name
  FROM {user_name |PUBLIC |role_name};

❖ ***<u>ABOUT THE PROJECT:-</u>***

- The project is basically about making a database for Restaurant billing system using important entities and attributes that are essential for making a simple database.
- The entities taken and the attributes are as follows:

- Customer: Cust_id (Primary key), Cust_name, Contact_no.
- Order: Order_no (Primary key), Cust_id (Foreign key), No_of_items.
- Restaurant: R_id (Primary key), Item_id, Order_no, r_name.
- Items: Item_id (Primary key), Name, No_of_items.
- Item_Category: Name (Primary key), Type.

➢ We have taken all the essential details that are required to make a database and used it as entities.

6. ***Actual Resources Used:-***

S. No.	***Name of Resource/material***	***Specifications***	***Qty***
1	Microsoft Word.	Version 2009	1
2	Oracle 9i.	Version 2009 or above	1

7. ***Skill Developed / Learning outcome of this Micro-Project:-***

➢ Hence, we have learnt to work with popular database system like My SQL and also to manage the SQL Database.

➢ We have also learnt to use different SQL Commands like Create, Insert, Alter, Select, Update, Delete, View and Cursor in My SQL.

8. ***Applications of this Micro-Project:-***

➢ This Database can be used to store and maintain Day by day Transactions/Details about Customers and items of the Restaurant.

➢ Due to the automated process, manual work is reduced and also the data is secured in the database.

****