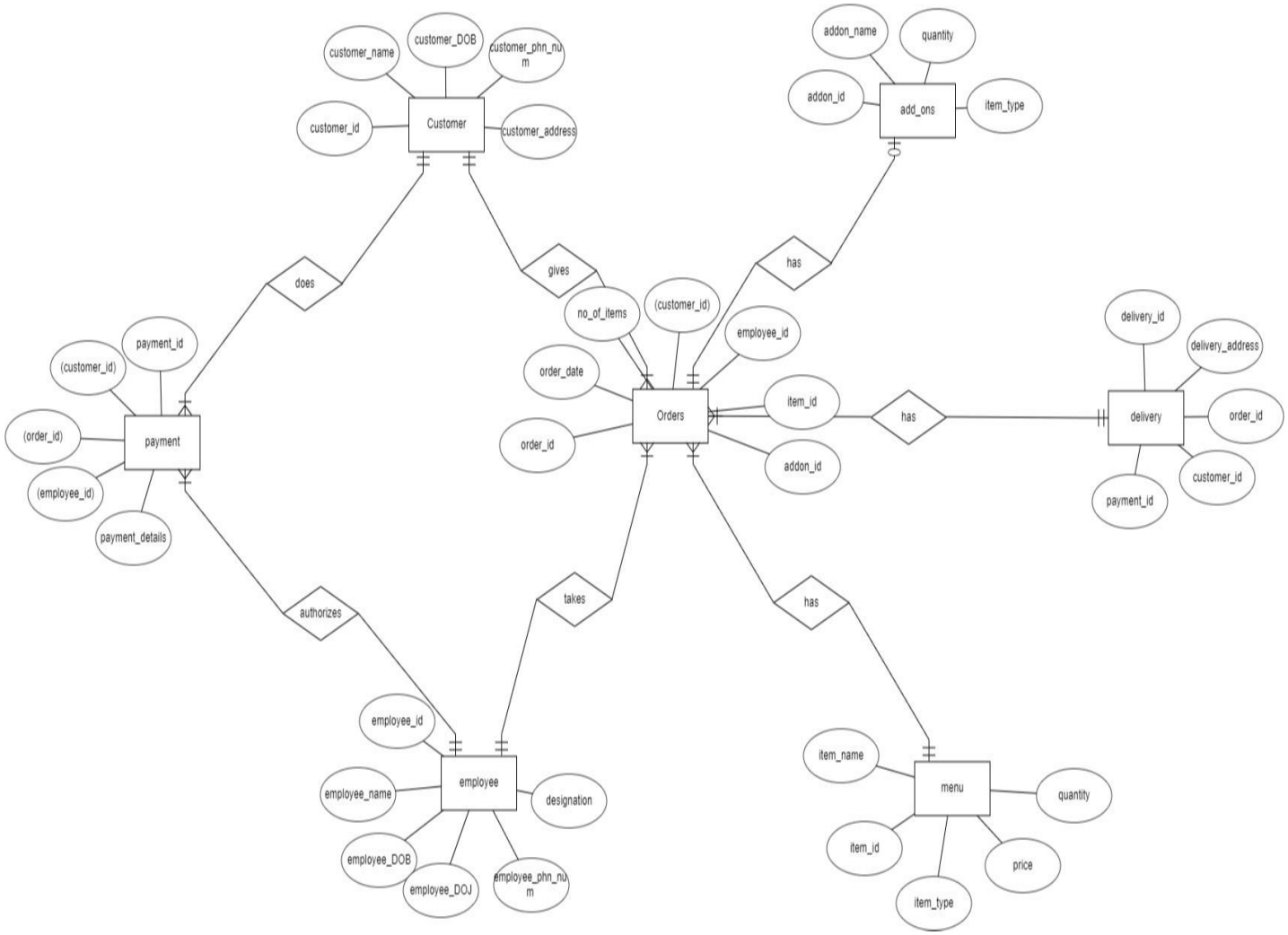
CDA 502: Database Management Systems

Project By

Jyoshna Pathapati (50469252)

Bulls Wings Data Management

Restaurant Management Database - Entity Relationship Diagram



Business Rules

Customer

- A customer can place one or more orders
- A customer can do one or more payments

Payment

- One or more payments can be done by one customer
- One or more payments can be recorded by one employee
- One payment can be done for one or more orders

Employee

- One employee can record one or more payments
- One employee can record one or more orders

Orders

- One or more orders can be placed by a customer
- One or more orders can be paid through one payment
- One or more orders can be recorded by an employee
- One order can have zero or many addons
- One or more orders can by delivered through a delivery
- One or more orders can have an item

Addons

- zero or more addons can be placed in an order

Delivery

- A delivery can have one or more orders

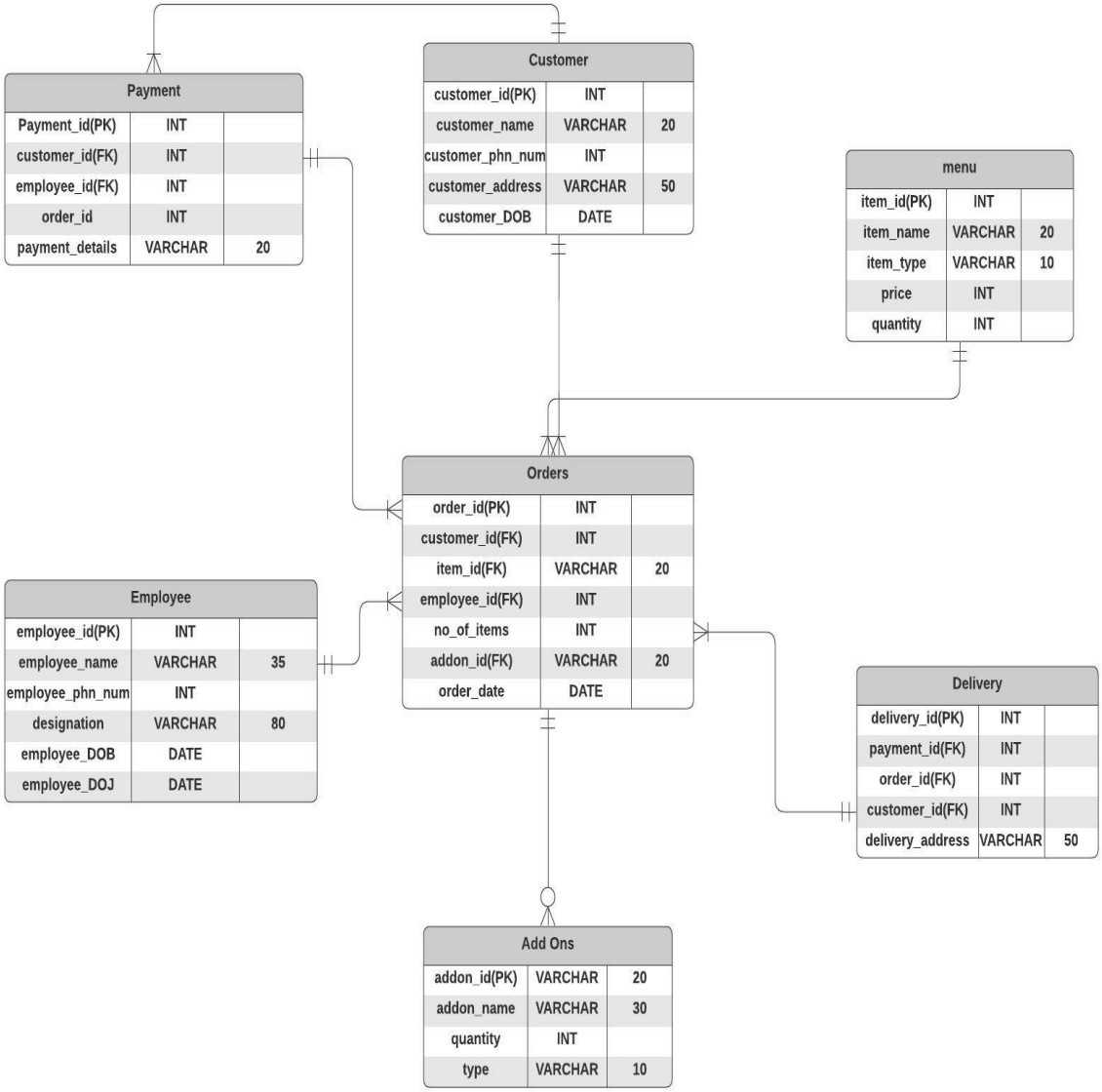
Item

- One item can be placed in one or more orders
- One or more items can be in one Menu

Menu

- One menu can have one or more items

Database Schema



Physical Design

```
#Creating Employee Table
CREATE  TABLE Employee
(
  employee_id INT NOT NULL,
  employee_name VARCHAR(95),
  employee_DOB DATE,
  employee_DOJ DATE,
  employee_phn_num INT ,
  designation VARCHAR(80),
  PRIMARY KEY (employee_id)
);
```

```
#Creating Add ons Table
CREATE  TABLE  Add_ons
(
  addon_id VARCHAR(20),
 addon_name VARCHAR(30),
 quantity INT NOT NULL,
 item_type VARCHAR(10),
 PRIMARY KEY (addon_id)
);
```

```
#Creating Menu Table
CREATE  TABLE Menu
(
  item_id INT NOT NULL,
  item_name VARCHAR(20),
  item_type VARCHAR(10),
  price INT ,
  quantity  INT,
 PRIMARY KEY (item_id)
);

#creating customer table
CREATE  TABLE  Customer
(
  customer_id INT NOT NULL,
  customer_name VARCHAR(20),
  customer_DOB DATE,
  customer_phn_num INT ,
  customer_address VARCHAR(50),
  PRIMARY KEY (customer_id)
);

#Creating Orders Table
CREATE TABLE  Orders
(
  order_id INT NOT NULL,
  order_date DATE,
  no_of_items INT,
```

```
  customer_id INT NOT NULL,
  employee_id INT NOT NULL,
  item_id INT NOT NULL,
  addon_id VARCHAR(20),
  PRIMARY KEY (order_id),
  FOREIGN KEY (customer_id) REFERENCES Customer(customer_id),
  FOREIGN KEY (employee_id) REFERENCES Employee(employee_id),
  FOREIGN KEY (item_id) REFERENCES Menu(item_id),
  FOREIGN KEY (addon_id) REFERENCES Add_ons(addon_id)
);

#Creating Payment Table
CREATE  TABLE Payment
(
  payment_id INT NOT NULL,
  payment_details VARCHAR(20),
  customer_id INT NOT NULL,
  employee_id INT NOT NULL,
  order_id INT NOT NULL,
  PRIMARY KEY (payment_id),
  FOREIGN KEY(customer_id) REFERENCES Customer(customer_id),
  FOREIGN KEY (order_id) REFERENCES Orders(order_id),
  FOREIGN KEY (employee_id) REFERENCES Employee(employee_id)
);
```

```
#Creating Delivery Table
CREATE TABLE  Delivery
(
  delivery_id INT NOT NULL,
  delivery_address VARCHAR(50),
  order_id INT NOT NULL,
  customer_id INT NOT NULL,
  payment_id INT NOT NULL,
  PRIMARY KEY (delivery_id),
  FOREIGN KEY (customer_id) REFERENCES Customer(customer_id),
  FOREIGN KEY (payment_id) REFERENCES Payment(payment_id),
  FOREIGN KEY (order_id) REFERENCES Orders(order_id)
);


#insert employee
INSERT INTO Employee Values (001,"Lalith","1990-08-23","2022-03-01","256718321","Chef");
INSERT INTO Employee Values (002,"Jyoshna","1947-08-13","2022-01-01","486634278","Manager");
INSERT INTO Employee Values (003,"Monisha","1996-02-09","2022-04-23","252144141","Chef");
INSERT INTO Employee Values (004,"Hruthika","1994-04-28","2022-05-11","244416321","Supervisor");
INSERT INTO Employee Values (005,"Sanjay","1997-12-16","2022-05-20","313433546","Manager");
INSERT INTO Employee Values (006,"Sai","1991-06-12","2022-06-19","827362515","Assistant");
INSERT INTO Employee Values (007,"Raju","1987-05-03","2022-07-11","351414441","Delivery Agent");
INSERT INTO Employee Values (008,"Mahendiran","1987-02-09","2022-07-12","826142363","Delivery Agent");
INSERT INTO Employee Values (009,"Kancharla","1994-07-06","2022-07-11","282351534","Server");
INSERT INTO Employee Values (010,"Tarun","1996-12-09","2022-06-14","721735141","Server");
```

```
#insert Add_ons
INSERT INTO Add_ons Values("ADD001","Sambar",1,"Sides");
INSERT INTO Add_ons Values("ADD002","Filter Coffee",1,"Drink");
INSERT INTO Add_ons Values("ADD003","Chutney",1,"Sides");
INSERT INTO Add_ons Values("ADD004","Bonda",3,"Snack");
INSERT INTO Add_ons Values("ADD005","Kesari",1,"Dessert");
INSERT INTO Add_ons Values("ADD006","Payasam",1,"Dessert");
INSERT INTO Add_ons Values("ADD007","Badam Milk",2,"Drink");
INSERT INTO Add_ons Values("ADD008","Bajji",5,"Snack");
INSERT INTO Add_ons Values("ADD009","Tea",1,"Drink");
INSERT INTO Add_ons Values("ADD010","Coke",1,"Drink");

#insert Menu
INSERT INTO Menu Values(001,"Idly","Tiffin",15,2);
INSERT INTO Menu Values(002,"Dosa","Tiffin",30,1);
INSERT INTO Menu Values(003,"Poori","Tiffin",40,2);
INSERT INTO Menu Values(004,"Chapathi","Tiffin",50,2);
INSERT INTO Menu Values(005,"Upma","Tiffin",60,1);
INSERT INTO Menu Values(006,"Rasam Rice","Lunch",50,1);
INSERT INTO Menu Values(007,"Curd Rice","Lunch",70,1);
INSERT INTO Menu Values(008,"Soup","Snack",150,1);
INSERT INTO Menu Values(009,"Chat","Snack",90,1);
INSERT INTO Menu Values(010,"Parota","Dinner",80,1);
```

#insert Customer

```
INSERT INTO Customer Values(01, "Aravind","1999-09-09", 23456789, "4121 Bailey Ave");
INSERT INTO Customer Values(02, "Varun", "1997-03-21",98765432, "4132 Bailey Ave");
INSERT INTO Customer Values(03, "Roshan","2003-05-30" ,567894321, "53 Tyler");
INSERT INTO Customer Values(04, "Neha", "1995-04-17",234567890, "14 Merimac");
INSERT INTO Customer Values(05, "Bindu", "1992-06-05",7890643, "63 Tyler");
INSERT INTO Customer Values(06, "Samantha","2000-03-19" ,34567892, "4252 Bailey Ave");
INSERT INTO Customer Values(07, "Smruthi","1998-04-15" ,45678322, "5172 Bailey Ave");
INSERT INTO Customer Values(08, "Rashmika","1995-09-25" ,56789432, "66 Tyler");
INSERT INTO Customer Values(09, "Prabhas", "1999-03-27",78965432, "78 Tyler");
INSERT INTO Customer Values(10, "Yashwanth","2001-02-02", 89076543, "15 Merimac");
```

#insert Order

```
INSERT INTO Orders Values(001,'2021-03-11',2,01,002,001,"ADD001");
INSERT INTO Orders Values(002,"2022-03-25",1,03,002,002,"ADD002");
INSERT INTO Orders Values(003,"2021-03-18",3,03,004,003,"ADD003");
INSERT INTO Orders Values(004,"2022-07-11",1,04,004,004,"ADD004");
INSERT INTO Orders Values(005,"2022-07-08",2,05,005,005,"ADD005");
INSERT INTO Orders Values(006,"2021-11-20",2,02,006,006,"ADD006");
INSERT INTO Orders Values(007,"2021-09-08",1,07,006,007,"ADD007");
INSERT INTO Orders Values(008,"2021-11-14",1,08,006,009,"ADD008");
INSERT INTO Orders Values(009,"2021-04-27",3,09,006,008,"ADD010");
INSERT INTO Orders Values(010,"2022-02-25",3,10,006,010,"ADD009");
INSERT INTO Orders Values(011,"2022-05-24",2,01,002,001,"ADD001");
INSERT INTO Orders Values(012,"2022-08-11",1,03,002,002,"ADD002");
INSERT INTO Orders Values(013,"2021-12-12",3,03,004,003,"ADD003");
INSERT INTO Orders Values(014,"2022-01-09",1,04,004,004,"ADD004");
```

```
INSERT INTO Orders Values(015,"2022-05-12",2,05,005,005,"ADD005");
INSERT INTO Orders Values(016,"2021-04-28",2,02,006,006,"ADD006");
INSERT INTO Orders Values(017,"2021-04-25",2,05,005,005,"ADD005");
INSERT INTO Orders Values(018,"2021-07-05",2,02,006,006,"ADD006");
INSERT INTO Orders Values(019,"2022-05-22",1,07,006,007,"ADD007");
INSERT INTO Orders Values(020,"2021-08-29",1,08,006,009,"ADD008");
INSERT INTO Orders Values(021,null,2,02,003,006,"ADD003");
INSERT INTO Orders Values(022,null,3,04,002,005,"ADD005");
INSERT INTO Orders Values(023,null,1,05,004,002,"ADD008");

#insert payment
INSERT INTO Payment Values (1,'Gpay',1,2,1);
INSERT INTO Payment Values (2,'Cash',3,2,2);
INSERT INTO Payment Values (3,'Phonepe',3,4,3);
INSERT INTO Payment Values (4,'Apple pay',4,4,4);
INSERT INTO Payment Values (5,'Paytm',5,5,5);
INSERT INTO Payment Values (6,'Cash',2,6,6);
INSERT INTO Payment Values (7,'Gpay',7,6,7);
INSERT INTO Payment Values (8,'Applepay',8,6,8);
INSERT INTO Payment Values (9,'Applepay',9,6,9);
INSERT INTO Payment Values (10,'Phonepe',10,6,10);
INSERT INTO Payment Values (11,'Gpay',1,2,11);
INSERT INTO Payment Values (12,'Cash',3,2,12);
INSERT INTO Payment Values (13,'Phonepe',3,4,13);
INSERT INTO Payment Values (14,'Apple pay',4,4,14);
INSERT INTO Payment Values (15,'Paytm',5,5,15);
INSERT INTO Payment Values (16,'Cash',2,6,16);
INSERT INTO Payment Values (17,'Gpay',5,5,17);
```

```
INSERT INTO Payment Values (18,'Applepay',2,6,18);
INSERT INTO Payment Values (19,'Applepay',7,6,19);
INSERT INTO Payment Values (20,'Phonepe',8,6,20);
```

```
#insert Delivery
INSERT INTO Delivery Values(1,"4121 Bailey Avenue",001,01,15);
INSERT INTO Delivery Values(2,"4252 Bailey Avenue",006,06,10);
INSERT INTO Delivery Values(3,"5172 Bailey Avenue",007,07,11);
INSERT INTO Delivery Values(4,"4132 Bailey Avenue",002,02,16);
INSERT INTO Delivery Values(5,"15 Merimac",010,10,19);
INSERT INTO Delivery Values(6,"14 Merimac",004,04,12);
INSERT INTO Delivery Values(7,"78 Tyler",009,09,18);
INSERT INTO Delivery Values(8,"66 Tyler",008,08,14);
INSERT INTO Delivery Values(9,"53 Tyler",003,03,13);
INSERT INTO Delivery Values(10,"63 Tyler",005,05,17);
```

Display Table Queries:

 # Add ons Table

Select * from add_ons;

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

addon_id	addon_name	quantity	item_type
ADD001	Sambar	1	Sides
ADD002	Filter Coffee	1	Drink
ADD003	Chutney	1	Sides
ADD004	Bonda	3	Snack
ADD005	Kesari	1	Dessert
ADD006	Payasam	1	Dessert
ADD007	Badam Milk	2	Drink
ADD008	Bajji	5	Snack
ADD009	Tea	1	Drink
ADD010	Coke	1	Drink
NULL	NULL	NULL	NULL

add_ons 188

 # Customer Table

Select * from customer;

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

customer_id	customer_name	customer_DOB	customer_phn_num	customer_address
1	Aravind	1999-09-09	23456789	4121 Bailey Ave
2	Varun	1997-03-21	98765432	4132 Bailey Ave
3	Roshan	2003-05-30	567894321	53 Tyler
4	Neha	1995-04-17	234567890	14 Merimac
5	Bindu	1992-06-05	7890643	63 Tyler
6	Samantha	2000-03-19	34567892	4252 Bailey Ave
7	Smruthi	1998-04-15	45678322	5172 Bailey Ave
8	Rashmika	1995-09-25	56789432	66 Tyler
9	Prabhas	1999-03-27	78965432	78 Tyler
10	Yashwanth	2001-02-02	89076543	15 Merimac
NULL	NULL	NULL	NULL	NULL

customer 189

 # Delivery Table

Select * from delivery;

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

delivery_id	delivery_address	order_id	customer_id	payment_id
1	4121 Bailey Avenue	1	1	15
2	4252 Bailey Avenue	6	6	10
3	5172 Bailey Avenue	7	7	11
4	4132 Bailey Avenue	2	2	16
5	15 Merimac	10	10	19
6	14 Merimac	4	4	12
7	78 Tyler	9	9	18
8	66 Tyler	8	8	14
9	53 Tyler	3	3	13
10	63 Tyler	5	5	17
NULL	NULL	NULL	NULL	NULL

delivery 190

 # Employee Table

Select * from employee;

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

employee_id	employee_name	employee_DOB	employee_DOJ	employee_phn_num	designation
1	Lalith	1990-08-23	2022-03-01	256718321	Chef
2	Jyoshna	1947-08-13	2022-01-01	486634278	Manager
3	Monisha	1996-02-09	2022-04-23	252144141	Chef
4	Hruthika	1994-04-28	2022-05-11	244416321	Supervisor
5	Sanjay	1997-12-16	2022-05-20	313433546	Manager
6	Sai	1991-06-12	2022-06-19	827362515	Assistant
7	Raju	1987-05-03	2022-07-11	351414441	Delivery Agent
8	Mahendiran	1987-02-09	2022-07-12	826142363	Delivery Agent
9	Kancharla	1994-07-06	2022-07-11	282351534	Server
10	Tarun	1996-12-09	2022-06-14	721735141	Server
NULL	NULL	NULL	NULL	NULL	NULL

employee 191

 # Menu Table

Select * from menu;

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

item_id	item_name	item_type	price	quantity
1	Idly	Tiffin	15	2
2	Dosa	Tiffin	30	1
3	Poori	Tiffin	40	2
4	Chapathi	Tiffin	50	2
5	Upma	Tiffin	60	1
6	Rasam Rice	Lunch	50	1
7	Curd Rice	Lunch	70	1
8	Soup	Snack	150	1
9	Chat	Snack	90	1
10	Parota	Dinner	80	1
NULL	NULL	NULL	NULL	NULL

menu 192

#Order Table

Select * from orders;

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

order_id	order_date	no_of_items	customer_id	employee_id	item_id	addon_id
1	2021-03-11	2	1	2	1	ADD001
2	2022-03-25	1	3	2	2	ADD002
3	2021-03-18	3	3	4	3	ADD003
4	2022-07-11	1	4	4	4	ADD004
5	2022-07-08	2	5	5	5	ADD005
6	2021-11-20	2	2	6	6	ADD006
7	2021-09-08	1	7	6	7	ADD007
8	2021-11-14	1	8	6	9	ADD008
9	2021-04-27	3	9	6	8	ADD010
10	2022-02-25	3	10	6	10	ADD009
11	2022-05-24	2	1	2	1	ADD001
12	2022-08-11	1	3	2	2	ADD002
13	2021-12-12	3	3	4	3	ADD003
14	2022-01-09	1	4	4	4	ADD004
15	2022-05-12	2	5	5	5	ADD005
16	2021-04-28	2	2	6	6	ADD006
17	2021-04-25	2	5	5	5	ADD005
18	2021-07-05	2	2	6	6	ADD006
19	2022-05-22	1	7	6	7	ADD007
20	2021-08-29	1	8	6	9	ADD008
21	NULL	2	2	3	6	ADD003
22	NULL	3	4	2	5	ADD005
23	NULL	1	5	4	2	ADD008
NULL	NULL	NULL	NULL	NULL	NULL	NULL

orders 193

 # Payment Table

Select * from payment;

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

payment_id	payment_details	customer_id	employee_id	order_id
1	Gpay	1	2	1
2	Cash	3	2	2
3	Phonepe	3	4	3
4	Apple pay	4	4	4
5	Paytm	5	5	5
6	Cash	2	6	6
7	Gpay	7	6	7
8	Applepay	8	6	8
9	Applepay	9	6	9
10	Phonepe	10	6	10
11	Gpay	1	2	11
12	Cash	3	2	12
13	Phonepe	3	4	13
14	Apple pay	4	4	14
15	Paytm	5	5	15
16	Cash	2	6	16
17	Gpay	5	5	17
18	Applepay	2	6	18
19	Applepay	7	6	19
20	Phonepe	8	6	20
NULL	NULL	NULL	NULL	NULL

payment 194

Populated data and query output

Query 1: Employees whose name starts with S who billed more than two orders

```
Select E.employee_id as Employee_ID,
        E.employee_name as Employee_Name,
        count(P.payment_id) as No_of_transactions_handled
From Employee as E
JOIN Orders as O on E.employee_id = O.employee_id
JOIN Payment as P on O.order_id = P.payment_id
Where E.employee_name Like "S%"
GROUP BY E.employee_id
Having Count(P.payment_id)>2;
```

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Employee_ID	Employee_Name	No_of_transactions_handled
5	Sanjay	3
6	Sai	9

Result 197

Query 2: Display summary of customers who paid using cash or apple pay and the type of item is Tiffin using string functions.

```
Select concat(upper(c.customer_name)," customer ID is  ",c.customer_id," and order ID is
",o.order_id," bought ",m.item_name," , item code being
",LPAD(RPAD(substr(m.item_name,1,3),6,'#'),9,'#'), " was priced at $",m.price) as
String_Function_Statement

From customer c

JOIN orders o on c.customer_id=o.customer_id

JOIN menu m on o.item_id=m.item_id

JOIN payment p on o.order_id=p.order_id

where p.payment_details="Cash" or p.payment_details="Applepay"  and m.item_type in ("Tiffin")

order by c.customer_name;
```

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

String_Function_Statement
ROSHAN customer ID is 3 and order ID is 2 bought Dosa , item code being ###Dos### was priced at $30
ROSHAN customer ID is 3 and order ID is 12 bought Dosa , item code being ###Dos### was priced at $30
VARUN customer ID is 2 and order ID is 6 bought Rasam Rice , item code being ###Ras### was priced at $50
VARUN customer ID is 2 and order ID is 16 bought Rasam Rice , item code being ###Ras### was priced at $50

Result 200

Query 3: Display top 3 customer details who placed highest number of orders

```
Select c.customer_name as Customer_Name,
        c.customer_id as Customer_ID,
        o.order_id as Order_ID,
        a.quantity as Quantity,
        a.addon_id as Addon_ID,
        Rank() Over(Order by a.quantity desc) as Rank_based_on_quantity
From customer c
JOIN orders o on c.customer_id=o.customer_id
JOIN add_ons a on a.addon_id=o.addon_id
GROUP BY c.customer_name
Limit 3;
```

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Customer_Name	Customer_ID	Order_ID	Quantity	Addon_ID	Rank_based_on_quantity
Rashmika	8	8	5	ADD008	1
Neha	4	4	3	ADD004	2
Smruthi	7	7	2	ADD007	3

Result 201

Query 4: Display the customer details who have paid highest amount for the orders on an average.

```
Select c.customer_id as Customer_ID,
        c.customer_name as Customer_Name,
        TRUNCATE(AVG(m.price),2) as Average_Amount,
        COUNT(o.order_id) as No_of_orders,
        o.no_of_items as No_of_items
from customer c
JOIN orders o on c.customer_id = o.customer_id
JOIN menu m on o.item_id=m.item_id
where c.customer_id = (Select c.customer_id from  customer c
                        JOIN orders o on c.customer_id = o.customer_id
                        JOIN menu m on o.item_id=m.item_id
                        GROUP BY c.customer_id
                      order by AVG(m.price) desc
                      limit 1);
```

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Customer_ID	Customer_Name	Average_Amount	No_of_orders	No_of_items
9	Prabhas	150.00	1	3

Result 202

Query 5: Display purchase history and the number of employees they have interacted with.

```
Select c.customer_id as Customer_ID,
        c.customer_name as Customer_Name,
        SUM(m.price) as Total_Amount,
        COUNT (DISTINCT o.order_id) as No_of_orders,
        COUNT (DISTINCT p.employee_id) as No_of_employees_interacted_with
from customer c
JOIN orders o on c.customer_id = o.customer_id
JOIN menu m on o.item_id=m.item_id
JOIN payment p on o.order_id = p.order_id
GROUP BY c.customer_id
order by sum(m.price) desc;
```

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Customer_ID	Customer_Name	Total_Amount	No_of_orders	No_of_employees_interacted_with
5	Bindu	180	3	1
8	Rashmika	180	2	1
2	Varun	150	3	1
9	Prabhas	150	1	1
3	Roshan	140	4	2
7	Smruthi	140	2	1
4	Neha	100	2	1
10	Yashwanth	80	1	1
1	Aravind	30	2	1

Result 205

Query 6: Display employee's employment details i.e, no of months of employment, no of transactions handled and rank them based on the no of transactions. Displaying the total amount handled along with the rank.

```
Select e.employee_name as Employee_Name,
        e.employee_id as Employee_ID,
        e.designation as Designation,
        employee_DOJ as Date_of_JOINing,
        (TIMESTAMPDIFF( month,employee_DOJ,sysdate())) as No_of_months_of_employement,
        COUNT(p.payment_id) as No_of_transactions_handled ,
        dense_rank() over (partition by e.designation order by count(p.payment_id) desc ) as
        Rank_based_on_no_of_payments_handled,
        SUM(m.price) as Bill_Amount
FROM employee e
JOIN orders o on e.employee_id = o.employee_id
JOIN payment p on o.order_id = p.order_id
JOIN menu m on m.item_id = o.item_id
GROUP BY e.employee_name
order by Rank_based_on_no_of_payments_handled;
```

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Employee_Name	Employee_ID	Designation	Date_of_Joining	No_of_months_of_employement	No_of_transactions_handled	Rank_based_on_no_of_payments_handled	Bill_Amount
Sai	6	Assistant	2022-06-19	5	9	1	700
Jyoshna	2	Manager	2022-01-01	10	4	1	90
Hruthika	4	Supervisor	2022-05-11	6	4	1	180
Sanjay	5	Manager	2022-05-20	6	3	2	180

Result 206

Query 7: Displaying all the orders which don't have an order date and by using Coalesce function replacing null values with 'Not Recorded' text. Displaying the maximum and minimum price of each order whose order date is not recorded.

```
SELECT c.customer_name as Customer_Name,
        m.price as Price,
        o.order_date as Order_Date,
        COALESCE(o.order_date,"Not Recorded") as Order_date_Rewritten
FROM customer c
Right OUTER JOIN orders o on c.customer_id=o.customer_id
JOIN menu m on m.item_id=o.item_id
where o.order_date is NULL;
```

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Customer_Name	Price	Order_Date	Order_date_Rewritten
Varun	50	NULL	Not Recorded
Neha	60	NULL	Not Recorded
Bindu	30	NULL	Not Recorded

Result 207

```
SELECT c.customer_name as Customer_Name,
        max(m.price) as Price,
        o.order_date as Order_Date,
        COALESCE (o.order_date,"Not Recorded") as Order_date_Rewritten
FROM customer c
Right OUTER JOIN orders o on c.customer_id=o.customer_id
JOIN menu m on m.item_id=o.item_id
where o.order_date is NULL;
```

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	Customer_Name	Price	Order_Date	Order_date_Rewritten
▶	Varun	60	NULL	Not Recorded

Result 208 ×

```
SELECT c.customer_name as Customer_Name,
        MIN(m.price) as Price,
        o.order_date as Order_Date,
        COALESCE (o.order_date,"Not Recorded") as Order_date_Rewritten
FROM customer c
Right OUTER JOIN orders o on c.customer_id=o.customer_id
JOIN menu m on m.item_id=o.item_id
where o.order_date is NULL;
```

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Customer_Name	Price	Order_Date	Order_date_Rewritten
Varun	30	NULL	Not Recorded

Result 209