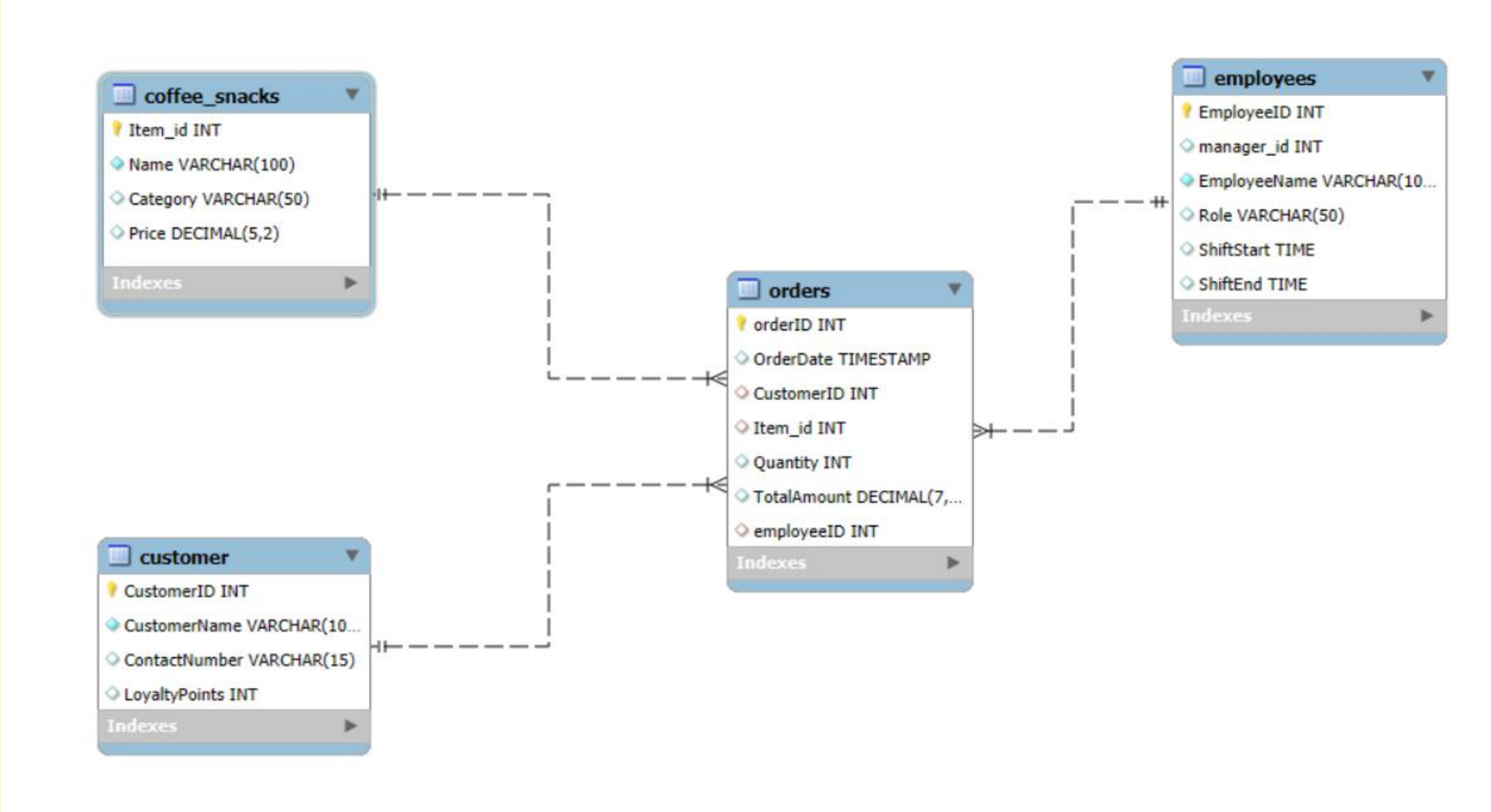
Coffee Shop Management System

INTRODUCTION

The Coffee Shop Management System will help the owner to extract valuable insights from the data like order summary, customer order details, top product sale wise, etc. using SQL queries. It will make their task quick and simple, saving the time for focusing on making improvements or enhancements required in increasing the profitability of the business.

Table of contents

ER Diagram


coffee_snacks
Item_id INT
Name VARCHAR(100)
Category VARCHAR(50)
Price DECIMAL(5,2)
Indexes
employees
EmployeeID INT
manager_id INT
EmployeeName VARCHAR(10...
Role VARCHAR(50)
ShiftStart TIME
ShiftEnd TIME
Indexes
orders
orderID INT
OrderDate TIMESTAMP
CustomerID INT
Item_id INT
Quantity INT
TotalAmount DECIMAL(7,...
employeeID INT
Indexes
customer
CustomerID INT
CustomerName VARCHAR(10...
ContactNumber VARCHAR(15)
LoyaltyPoints INT
Indexes

Tables

Coffee_snacks table

The table shows the list of coffees and snacks served, their id and price.

Description desc coffee_snacks;

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

Field	Type	Null	Key	Default	Extra
Item_id	int	NO	PRI	NULL	auto_increment
Name	varchar(100)	NO		NULL	
Category	varchar(50)	YES		NULL	
Price	decimal(5,2)	YES		NULL	

Contents select * from coffee_snacks;

Result Grid | Filter Rows:

Item_id	Name	Category	Price
1	Espresso	Beverage	2.50
2	Latte	Beverage	3.50
3	Cappuccino	Beverage	3.00
4	Americano	Beverage	2.75
5	Mocha	Beverage	3.75
6	Green Tea	Beverage	2.25
7	Black Tea	Beverage	2.00
8	Chai Latte	Beverage	3.25
9	Iced Coffee	Beverage	3.00
10	Hot Chocolate	Beverage	3.50
11	Blueberry Mu...	Snack	2.50
12	Croissant	Snack	2.75
13	Bagel	Snack	2.00
14	Granola Bar	Snack	1.50
15	Chocolate Ch...	Snack	1.75
16	Sandwich	Food	5.50
17	Salad	Food	4.75
18	Pasta	Food	6.00
19	Soup	Food	3.95
20	Fruit Bowl	Food	4.00
NULL	NULL	NULL	NULL

coffee_snacks 2

Orders table

The table shows details of orders placed like orderId, quantity and total amount.

Description desc orders;

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

Field	Type	Null	Key	Default	Extra
orderID	int	NO	PRI	NULL	auto_increment
OrderDate	timestamp	YES		CURRENT_TIMESTAMP	DEFAULT_GENERATED
CustomerID	int	YES	MUL	NULL	
Item_id	int	YES	MUL	NULL	
Quantity	int	YES		NULL	
TotalAmount	decimal(7,2)	YES		NULL	

Contents select * from orders;

Result Grid | Filter Rows: | Edit: | Export/Impo

orderID	OrderDate	CustomerID	Item_id	Quantity	TotalAmount
1	2025-01-01 10:30:00	1	2	1	5.50
2	2025-01-02 11:00:00	2	3	2	10.25
3	2025-01-03 09:45:00	4	1	1	7.75
4	2025-01-04 14:15:00	5	4	3	12.50
5	2025-01-05 08:20:00	3	6	1	4.00
6	2025-01-06 10:00:00	7	7	2	8.75
7	2025-01-07 12:30:00	8	9	1	15.50
8	2025-01-08 13:10:00	6	12	1	9.25
9	2025-01-09 15:45:00	9	11	1	5.00
10	2025-01-10 11:20:00	10	13	3	14.75
11	2025-01-11 14:30:00	11	5	2	3.75
12	2025-01-12 10:50:00	12	8	1	6.00
13	2025-01-13 13:15:00	15	17	1	8.25
14	2025-01-14 09:40:00	16	18	2	7.00
15	2025-01-15 11:55:00	13	19	1	11.00
16	2025-01-16 12:45:00	14	14	1	10.50
17	2025-01-17 14:10:00	18	10	1	9.75
18	2025-01-18 08:35:00	17	20	1	4.50
19	2025-01-19 10:15:00	20	15	2	12.00
20	2025-01-20 13:30:00	19	16	1	14.25
NULL	NULL	NULL	NULL	NULL	NULL

orders 1

Customer table

The table shows details of customers like customerId, name and loyalty points.

Description desc customer;

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

Field	Type	Null	Key	Default	Extra
CustomerID	int	NO	PRI	NULL	auto_increment
CustomerName	varchar(100)	NO		NULL	
ContactNumber	varchar(15)	YES		NULL	
LoyaltyPoints	int	YES		0	

Contents select * from customer;

Result Grid | Filter Rows: | Edit:

CustomerID	CustomerName	ContactNumber	LoyaltyPoints
1	Alice Johnson	555-1234	150
2	Bob Smith	555-5678	200
3	Charlie Brown	555-9876	50
4	Diana Prince	555-4321	300
5	Ethan Hunt	555-6789	180
6	Fiona Harper	555-8765	90
7	George Martin	555-1111	250
8	Hannah Lee	555-2222	70
9	Ian Black	555-3333	40
10	Jane Foster	555-4444	120
11	Kyle Richards	555-5555	60
12	Laura Wilson	555-6666	130
13	Mike Davis	555-7777	220
14	Nina Gomez	555-8888	80
15	Oscar Wilde	555-9999	110
16	Paula White	555-0000	170
17	Quinn Parker	555-1010	140
18	Rachel Adams	555-2020	190
19	Steve Rogers	555-3030	210
20	Tina Turner	555-4040	160
NULL	NULL	NULL	NULL

customer 3

Employees table

The table shows details of employees like id, name, role and shift details.

Description desc employees;

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

Field	Type	Null	Key	Default	Extra
EmployeeID	int	NO	PRI	NULL	auto_increment
EmployeeName	varchar(100)	NO		NULL	
Role	varchar(50)	YES		NULL	
ShiftStart	time	YES		NULL	
ShiftEnd	time	YES		NULL	

Contents select * from employees;

Result Grid | Filter Rows: | Edit: | Export/Import:

EmployeeID	manager_id	EmployeeName	Role	ShiftStart	ShiftEnd
2	NULL	Sarah Connor	Manager	09:00:00	17:00:00
3	2	Tom Hardy	Cashier	10:00:00	16:00:00
4	2	Emma Watson	Barista	07:00:00	13:00:00
5	2	Chris Pratt	Chef	06:00:00	12:00:00
6	2	Jessica Alba	Cashier	11:00:00	17:00:00
7	13	David Beckham	Barista	13:00:00	19:00:00
8	NULL	Natalie Portman	Manager	08:30:00	16:30:00
9	8	Ryan Gosling	Barista	12:00:00	18:00:00
10	8	Scarlett Johansson	Chef	06:30:00	12:30:00
11	8	Hugh Jackman	Cashier	10:30:00	16:30:00
12	8	Anne Hathaway	Barista	09:00:00	15:00:00
13	NULL	Brad Pitt	Manager	11:00:00	19:00:00
14	13	Mila Kunis	Cashier	07:30:00	13:30:00
15	13	Leonardo DiCaprio	Chef	06:00:00	14:00:00
16	13	Zoe Saldana	Barista	08:00:00	14:00:00
17	NULL	Chris Hemsworth	Manager	10:00:00	18:00:00
18	17	Jennifer Lawrence	Chef	05:30:00	11:30:00
19	17	Matt Damon	Cashier	09:30:00	15:30:00
20	17	Emily Blunt	Barista	11:00:00	17:00:00
NULL	NULL	NULL	NULL	NULL	NULL

employees 26

Sub-Queries

Display top 3 selling items with their price.

Result Grid | Filter Rows:

name	price
Espresso	2.50
Latte	3.50
Cappuccino	3.00

Syntax:

```
select name, price from coffee_snacks
 where item_id in
(select item_id from orders group by item_id
order by sum(quantity))
limit 3;
```

Display the names of the employees who worked more than the average shift time.

Result Grid | Filter Rows: | Export:

EmployeeName	Role	ShiftStart	ShiftEnd
Sarah Connor	Manager	09:00:00	17:00:00
Natalie Portman	Manager	08:30:00	16:30:00
Brad Pitt	Manager	11:00:00	19:00:00
Leonardo DiCaprio	Chef	06:00:00	14:00:00
Chris Hemsworth	Manager	10:00:00	18:00:00

Syntax:

```
select EmployeeName, Role, ShiftStart, ShiftEnd
from Employees
where timediff(ShiftEnd, ShiftStart) >
(select avg(timediff(ShiftEnd, ShiftStart))
from Employees);
```

Display the names of the customers who didn't order from beverage category.

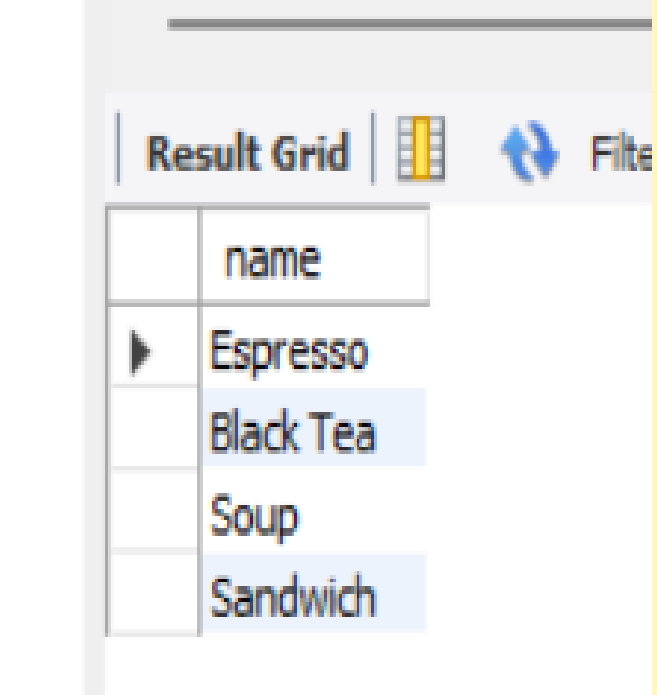
Result Grid | Filter Rows:

CustomerName
Fiona Harper
Ian Black
Jane Foster
Mike Davis
Nina Gomez
Oscar Wilde
Paula White
Quinn Parker
Steve Rogers
Tina Turner

Syntax:

```
select CustomerName from Customer
where CustomerID not in
(select distinct CustomerID  from orders
WHERE Item_ID in (select Item_ID from
coffee_snacks where Category = 'Beverage'));
```

Display the names of the items ordered by customers with more than 200 Loyalty Points.

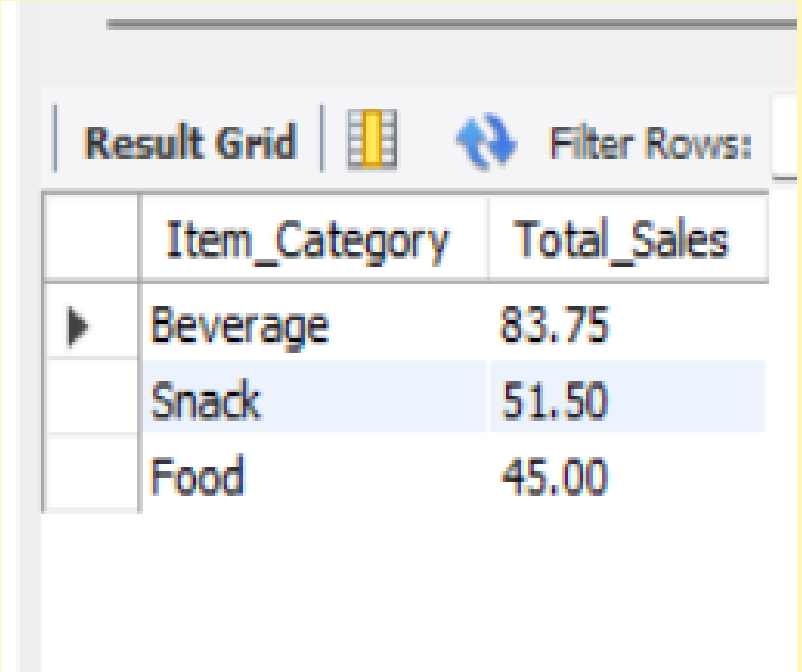


Syntax:

```
select name from coffee_snacks
 where item_id in
(select item_id from orders
where CustomerID in
(select CustomerID from customer
 where LoyaltyPoints > 200
)
);
```

Joins

Display total sales for each category.



Result Grid | Filter Rows:

Item_Category	Total_Sales
Beverage	83.75
Snack	51.50
Food	45.00

Syntax:

```
select cs.Category as
Item_Category,
sum(o.TotalAmount) as Total_Sales
from orders o join coffee_snacks cs
 on o.Item_id = cs.Item_id
group by cs.Category;
```

Display top 5 highest spending customers with their orders and total amount spent.

Result Grid | Filter Rows: | Export:

OrderID	CustomerName	ItemName	TotalAmount
7	Hannah Lee	Iced Coffee	15.50
10	Jane Foster	Bagel	14.75
20	Steve Rogers	Sandwich	14.25
4	Ethan Hunt	Americano	12.50
19	Tina Turner	Chocolate Chip Cookie	12.00

Syntax:

```
select Orders.OrderID,
Customer.CustomerName,
coffee_Snacks.Name as ItemName,
Orders.TotalAmount
from Orders INNER JOIN Customer
on Orders.CustomerID =
Customer.CustomerID
INNER JOIN Coffee_Snacks
on Orders.Item_id = Coffee_Snacks.Item_id
where Customer.CustomerID in
(selectTopCustomers.CustomerID    from
(select CustomerID from Orders  group by
CustomerID
order by sum(TotalAmount) desc
limit 5)
 as TopCustomers);
```

Display total sales handled by each employee.

Result Grid | Filter Rows:

EmployeeName	TotalSales
Chris Hemsworth	29.00
Scarlett Johansson	23.50
David Beckham	22.75
Mila Kunis	15.50
Chris Pratt	12.50
Matt Damon	12.00
Sarah Connor	10.50
Emma Watson	10.25
Ryan Gosling	9.25
Brad Pitt	8.75
Anne Hathaway	7.75
Natalie Portman	7.00
Hugh Jackman	6.00
Jessica Alba	5.50

Syntax:

```
select Employees.EmployeeName,
sum(Orders.TotalAmount) as
TotalSales
from
Orders INNER JOIN Employees
on Orders.employeeID =
Employees.EmployeeID
Group by
Employees.EmployeeName
order by TotalSales desc;
```

Display employee hierarchy.

Result Grid | Filter Rows:

manager	employee
Sarah Connor	John Doe
Sarah Connor	Tom Hardy
Sarah Connor	Emma Watson
Sarah Connor	Chris Pratt
Sarah Connor	Jessica Alba
Brad Pitt	David Beckham
Natalie Portman	Ryan Gosling
Natalie Portman	Scarlett Johansson
Natalie Portman	Hugh Jackman
Natalie Portman	Anne Hathaway
Brad Pitt	Mila Kunis
Brad Pitt	Leonardo DiCaprio
Brad Pitt	Zoe Saldana
Chris Hemsworth	Jennifer Lawrence
Chris Hemsworth	Matt Damon
Chris Hemsworth	Emily Blunt

Self Join

Syntax:

```
select e1.EmployeeName as
manager,
e2.EmployeeName as employee
from Employees e1 JOIN
Employees e2
ON e1.employeeid = e2.manager_ID;
```

Views

Daily Sales Report.

Result Grid | Filter Rows: | Exp

SaleDate	Day	TotalSales	TotalOrders
2025-01-01	Wednesday	5.50	1
2025-01-02	Thursday	10.25	1
2025-01-03	Friday	7.75	1
2025-01-04	Saturday	12.50	1
2025-01-05	Sunday	4.00	1
2025-01-06	Monday	8.75	1
2025-01-07	Tuesday	15.50	1
2025-01-08	Wednesday	9.25	1
2025-01-09	Thursday	5.00	1
2025-01-10	Friday	14.75	1
2025-01-11	Saturday	3.75	1
2025-01-12	Sunday	6.00	1
2025-01-13	Monday	8.25	1
2025-01-14	Tuesday	7.00	1
2025-01-15	Wednesday	11.00	1
2025-01-16	Thursday	10.50	1
2025-01-17	Friday	9.75	1
2025-01-18	Saturday	4.50	1
2025-01-19	Sunday	12.00	1
2025-01-20	Monday	14.25	1

Syntax:

```
create view DailySales as
select
DATE(OrderDate) as SaleDate,
DAYNAME(OrderDate) as Day,
SUM(TotalAmount) as TotalSales,
COUNT(OrderID) as TotalOrders
from Orders
group by SaleDate, Day
order by SaleDate;
```

Items sold with highest quantities in each category.

Result Grid | Filter Rows:

Category	Name	TotalQuantity
Beverage	Americano	3
Snack	Bagel	3
Food	Pasta	2

Syntax:

```
create view CategoryMaxQty as
with CategoryTotals as
(select
cs.category,cs.name,max(o.quantity) as
totalquantity from coffee_snacks cs left
join orders o on cs.item_id = o.item_id
group by cs.name,cs.category order by
totalquantity desc),
MaxTotals as
(select cs.category,max(o.quantity) as
totalquantity from coffee_snacks cs left
join orders o on cs.item_id = o.item_id
group by category)
select  ct.Category, ct.Name,
ct.TotalQuantity
from CategoryTotals ct JOIN MaxTotals
mct
on ct.Category = mct.Category AND
ct.TotalQuantity = mct.totalQuantity;
select * from CategoryMaxQty;
```