

COFFEE RECOMMENDATION SYSTEM

Objective:

The coffee recommendation system is to provide personalized and relevant coffee suggestions to users based on their preferences, tastes, and previous interactions with coffee products. Such systems are designed to enhance the user experience by offering tailored coffee choices, which can lead to increased customer satisfaction and loyalty for coffee businesses.

Goals:

- Personalization
- Improve User Engagement
- Enhanced Customer Experience
- Increase Sales and Revenue

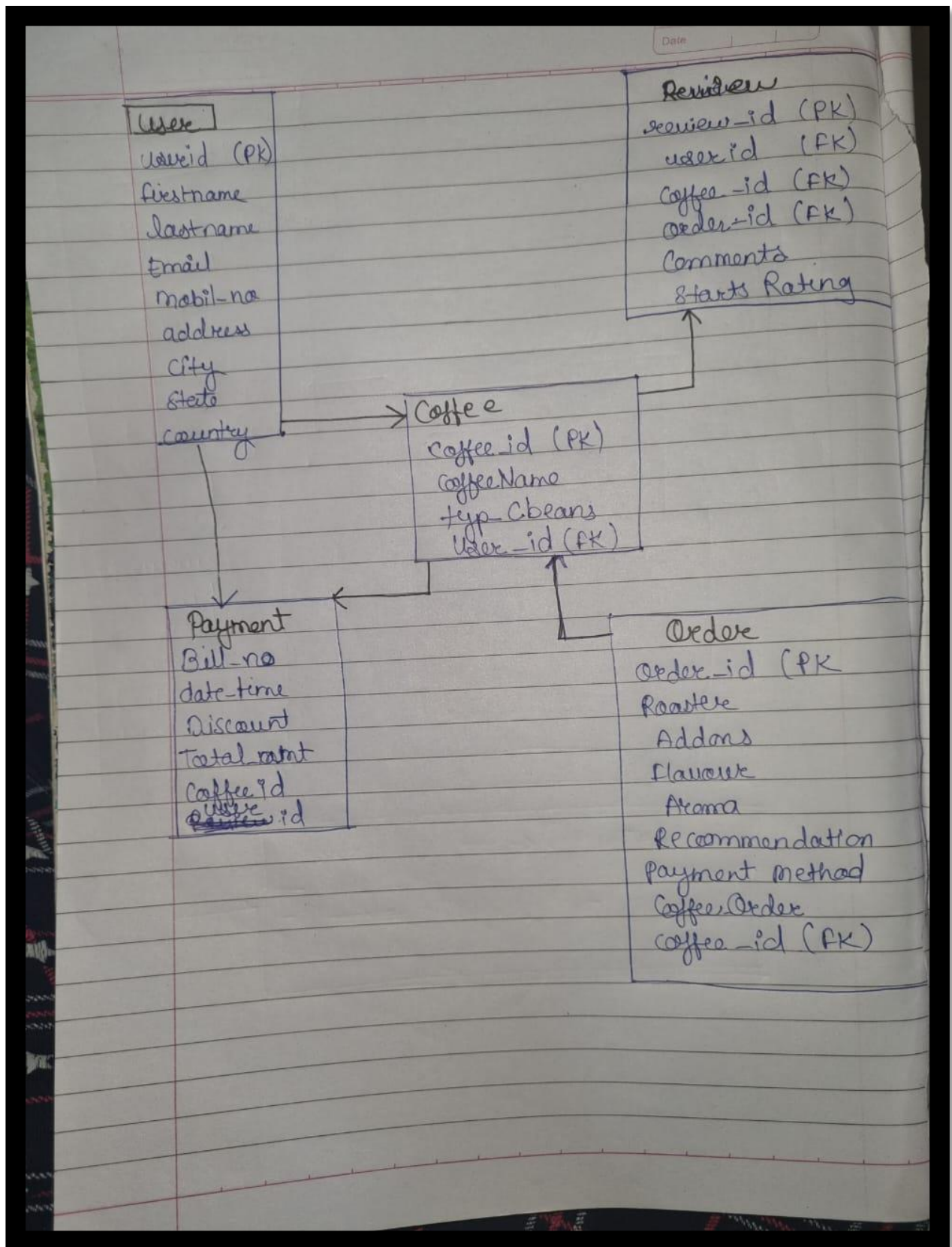
Future vision:

The future vision of coffee recommendation systems is promising, with advancements in AI, machine learning, voice interfaces, and IoT integration. These future developments will enable even more accurate, multimodal, and interactive recommendations, making the coffee experience more seamless and enjoyable for users. Additionally, an emphasis on ethical and sustainable coffee recommendations will align with the increasing demand for responsible consumer choices.

Conclusion:

In conclusion, coffee recommendation systems play a crucial role in enhancing the coffee discovery journey for users and driving growth in the coffee industry. These systems leverage artificial intelligence, machine learning, and user data to provide personalized and relevant coffee suggestions based on individual preferences, tastes, and behaviour. By understanding user preferences, such systems can offer tailored coffee choices, leading to increased customer satisfaction, engagement, and loyalty.

ER Diagram:



use dataset;

```
CREATE TABLE User( User_id INT PRIMARY KEY,  
    first_name VARCHAR(50) NOT NULL, last_name VARCHAR(50) NOT NULL,  
    Email VARCHAR(100) NOT NULL, Mobil_no VARCHAR(15) NOT NULL,  
    address VARCHAR(200),city VARCHAR(50),  
    state VARCHAR(50),country VARCHAR(50));
```

```
INSERT INTO user (user_id, first_name, last_name, email, Mobil_no, address, city, state, country)  
VALUES
```

```
(01, 'reeyansh', 'humne', 'reeyu@gmail.com', '12345890', 'jaipur', 'jaipur', 'Rajasthan', 'India'),
```

```
(02, 'priya', 'joshi', 'priyaj@gmail.com', '987210', '456 Park Ave', 'Los Angeles', 'CA', 'USA'),
```

```
(03, 'Kushal', 'badrike', 'kushal@gmail.com', '9515555', 'Sector16', 'Sanpada', 'Maharashtra', 'India'),
```

```
(04, 'bhau', 'kadam', 'bhau@gmail.com', '89842233', '321 Oak Blvd', 'Houston', 'TX', 'USA'),
```

```
(05, 'Vimlabai', 'More', 'vimla@gmail.com', '87954444', 'deccan', 'pune', 'Maharashtra', 'India');
```

```
select * from user;
```

The screenshot shows a SQL IDE interface with a menu bar (Database, Server, Tools, Scripting, Help) and a toolbar. The main editor displays the following SQL script:

```
1 • use dataset;  
2  
3 • CREATE TABLE User (  
4     User_id INT PRIMARY KEY,  
5     first_name VARCHAR(50) NOT NULL,  
6     last_name VARCHAR(50) NOT NULL,  
7     Email VARCHAR(100) NOT NULL,  
8     Mobil_no VARCHAR(15) NOT NULL,  
9     address VARCHAR(200),  
10    city VARCHAR(50),  
11    state VARCHAR(50),  
12    country VARCHAR(50)  
13 );  
14 • use dataset;  
15 • INSERT INTO user (user_id, first_name, last_name, email, Mobil_no, address, city, state, country)  
16 VALUES  
17 (01, 'reeyansh', 'humne', 'reeyu@gmail.com', '12345890', 'jaipur', 'jaipur', 'Rajasthan', 'India'),  
18 (02, 'priya', 'joshi', 'priyaj@gmail.com', '987210', '456 Park Ave', 'Los Angeles', 'CA', 'USA'),  
19 (03, 'Kushal', 'badrike', 'kushal@gmail.com', '9515555', 'Sector16', 'Sanpada', 'Maharashtra', 'India'),  
20 (04, 'bhau', 'kadam', 'bhau@gmail.com', '89842233', '321 Oak Blvd', 'Houston', 'TX', 'USA'),  
21 (05, 'Vimlabai', 'More', 'vimla@gmail.com', '87954444', 'deccan', 'pune', 'Maharashtra', 'India');  
22 • select * from user;  
23
```

Below the editor, the 'Result Grid' shows the data inserted into the 'user' table:

	User_id	first_name	last_name	Email	Mobil_no	address	city	state	country
1	reeyansh	humne	reeyu@gmail.com	12345890	jaipur	jaipur	Rajasthan	India	
2	priya	joshi	priyaj@gmail.com	987210	456 Park Ave	Los Angeles	CA	USA	
3	Kushal	badrike	kushal@gmail.com	9515555	Sector16	Sanpada	Maharashtra	India	
4	bhau	kadam	bhau@gmail.com	89842233	321 Oak Blvd	Houston	TX	USA	
5	Vimlabai	More	vimla@gmail.com	87954444	deccan	pune	Maharashtra	India	
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

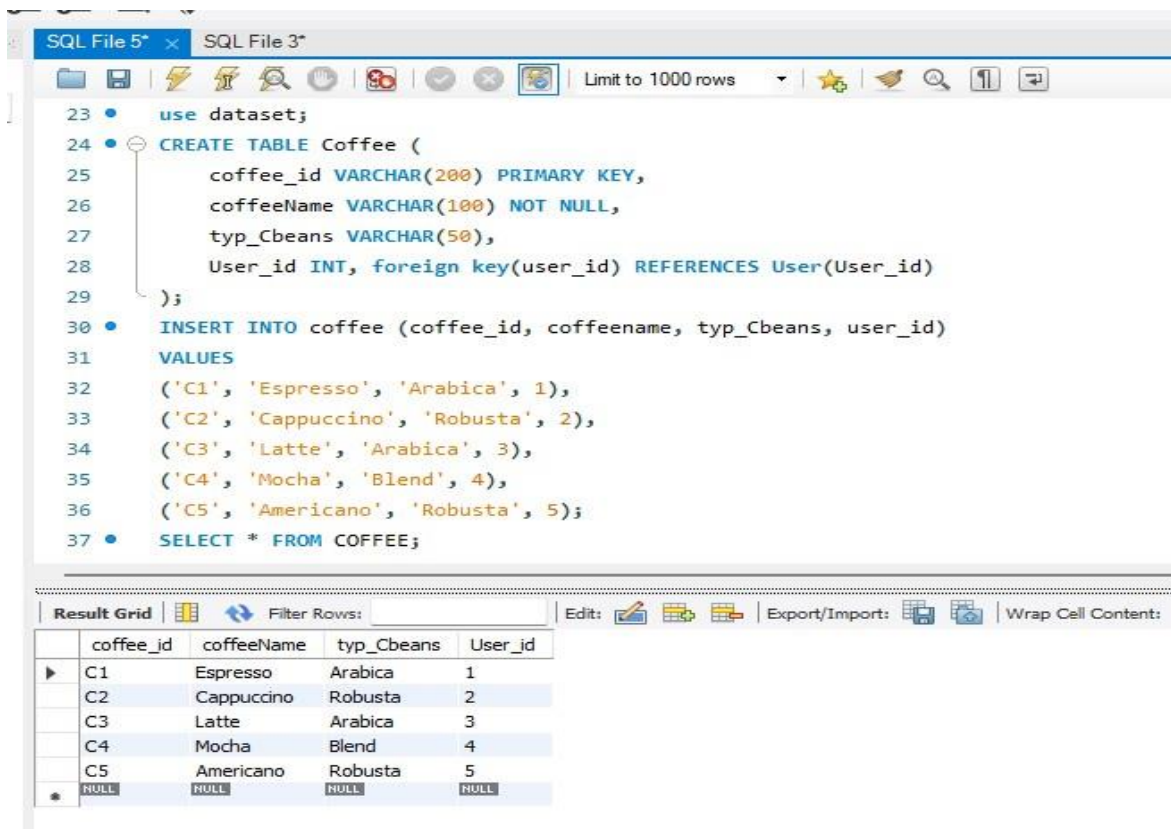
```

CREATE TABLE Coffee (
    coffee_id VARCHAR(200) PRIMARY KEY,
    coffeeName VARCHAR(100) NOT NULL,
    typ_Cbeans VARCHAR(50),
    User_id INT, foreign key(user_id) REFERENCES User(User_id)
);

INSERT INTO coffee (coffee_id, coffeename, typ_Cbeans, user_id)
VALUES
('C1', 'Espresso', 'Arabica', 1),
('C2', 'Cappuccino', 'Robusta', 2),
('C3', 'Latte', 'Arabica', 3),
('C4', 'Mocha', 'Blend', 4),
('C5', 'Americano', 'Robusta', 5);

SELECT * FROM COFFEE;

```



The screenshot shows a SQL IDE with two tabs: "SQL File 5*" and "SQL File 3*". The main editor displays the SQL code from the previous block. The interface includes a toolbar with icons for file operations, execution, and search. Below the editor, the "Result Grid" is visible, showing the output of the SQL queries. The grid has five columns: coffee_id, coffeeName, typ_Cbeans, and User_id. It contains five rows of data corresponding to the coffee types defined in the code. A sixth row shows NULL values for all columns.

	coffee_id	coffeeName	typ_Cbeans	User_id
▶	C1	Espresso	Arabica	1
	C2	Cappuccino	Robusta	2
	C3	Latte	Arabica	3
	C4	Mocha	Blend	4
	C5	Americano	Robusta	5
★	NULL	NULL	NULL	NULL

```
CREATE TABLE Orders ( order_id INT PRIMARY KEY, Coffee_id VARCHAR(200),
    FOREIGN KEY (coffee_id) REFERENCES coffee(coffee_id),
    Roaster VARCHAR(100), Addons VARCHAR(100), Flavours VARCHAR(50), Aroma VARCHAR(50),
    Recommendation VARCHAR(200), payment_Method varchar(100), Coffee_order VARCHAR(100));
```

```
INSERT INTO Orders (order_id, Coffee_id, Roaster, Addons, Flavours, Aroma,payment_Method,
Coffee_order,Recommendation)
```

VALUES

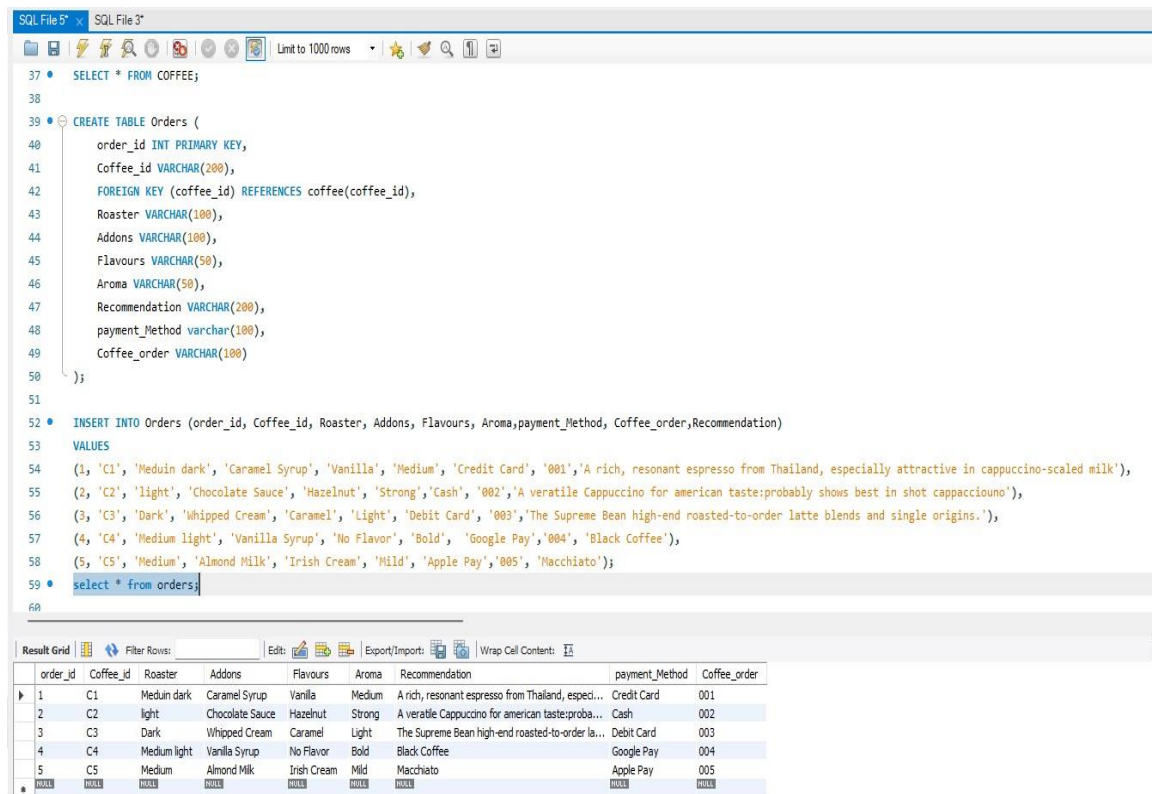
(1, 'C1', 'Meduin dark', 'Caramel Syrup', 'Vanilla', 'Medium', 'Credit Card', '001','A rich, resonant espresso from Thailand, especially attractive in cappuccino-scaled milk'),

(2, 'C2', 'light', 'Chocolate Sauce', 'Hazelnut', 'Strong','Cash', '002','A veratile Cappuccino for american taste:probably shows best in shot cappacciouno'),

(3, 'C3', 'Dark', 'Whipped Cream', 'Caramel', 'Light', 'Debit Card', '003','The Supreme Bean high-end roasted-to-order latte blends and single origins.'),

(4, 'C4', 'Medium light', 'Vanilla Syrup', 'No Flavor', 'Bold', 'Google Pay','004', 'Black Coffee'),

(5, 'C5', 'Medium', 'Almond Milk', 'Irish Cream', 'Mild', 'Apple Pay','005', 'Macchiato');



The screenshot shows a SQL editor window titled 'SQL File 3' with a toolbar and a 'Limit to 1000 rows' dropdown. The SQL code is as follows:

```
37 SELECT * FROM COFFEE;
38
39 CREATE TABLE Orders (
40     order_id INT PRIMARY KEY,
41     Coffee_id VARCHAR(200),
42     FOREIGN KEY (coffee_id) REFERENCES coffee(coffee_id),
43     Roaster VARCHAR(100),
44     Addons VARCHAR(100),
45     Flavours VARCHAR(50),
46     Aroma VARCHAR(50),
47     Recommendation VARCHAR(200),
48     payment_Method varchar(100),
49     Coffee_order VARCHAR(100)
50 );
51
52 INSERT INTO Orders (order_id, Coffee_id, Roaster, Addons, Flavours, Aroma,payment_Method, Coffee_order,Recommendation)
53 VALUES
54 (1, 'C1', 'Meduin dark', 'Caramel Syrup', 'Vanilla', 'Medium', 'Credit Card', '001','A rich, resonant espresso from Thailand, especially attractive in cappuccino-scaled milk'),
55 (2, 'C2', 'light', 'Chocolate Sauce', 'Hazelnut', 'Strong','Cash', '002','A veratile Cappuccino for american taste:probably shows best in shot cappacciouno'),
56 (3, 'C3', 'Dark', 'Whipped Cream', 'Caramel', 'Light', 'Debit Card', '003','The Supreme Bean high-end roasted-to-order latte blends and single origins.'),
57 (4, 'C4', 'Medium light', 'Vanilla Syrup', 'No Flavor', 'Bold', 'Google Pay','004', 'Black Coffee'),
58 (5, 'C5', 'Medium', 'Almond Milk', 'Irish Cream', 'Mild', 'Apple Pay','005', 'Macchiato');
59 select * from orders;
60
```

Below the code, the 'Result Grid' shows the data inserted into the 'Orders' table:

order_id	Coffee_id	Roaster	Addons	Flavours	Aroma	Recommendation	payment_Method	Coffee_order
1	C1	Meduin dark	Caramel Syrup	Vanilla	Medium	A rich, resonant espresso from Thailand, espec...	Credit Card	001
2	C2	light	Chocolate Sauce	Hazelnut	Strong	A veratile Cappuccino for american taste:proba...	Cash	002
3	C3	Dark	Whipped Cream	Caramel	Light	The Supreme Bean high-end roasted-to-order la...	Debit Card	003
4	C4	Medium light	Vanilla Syrup	No Flavor	Bold	Black Coffee	Google Pay	004
5	C5	Medium	Almond Milk	Irish Cream	Mild	Macchiato	Apple Pay	005


```

CREATE TABLE review (review_id INT PRIMARY KEY, user_id INT,
    FOREIGN KEY (user_id) REFERENCES user(user_id),
    coffee_id VARCHAR(200), FOREIGN KEY (coffee_id) REFERENCES coffee(coffee_id),
    order_id INT, FOREIGN KEY (order_id) REFERENCES orders(order_id),
    Comments VARCHAR(200), StarRating VARCHAR(5));
INSERT INTO review (review_id, user_id, coffee_id, order_id, Comments, StarRating)
VALUES
(101, 01, 'C1', 1, 'Great coffee, loved the flavor!', '5'),
(102, 02, 'C2', 2, 'Amazing cappuccino, very creamy.', '4.5'),
(103, 03, 'C3', 3, 'Excellent latte, will order again.', '5'),
(104, 04, 'C4', 4, 'Not satisfied with the black coffee.', '2'),
(105, 05, 'C5', 5, 'Delicious macchiato, perfect balance.', '4');
select * from review;

```

The screenshot shows a SQL IDE interface with a toolbar at the top and a code editor. The code editor contains the SQL commands from the previous block. Below the code editor is a 'Result Grid' tab showing the output of the 'select * from review;' query. The grid has 7 columns: review_id, user_id, coffee_id, order_id, Comments, and StarRating. It displays 5 rows of data corresponding to the inserted records, followed by a row of NULL values.

review_id	user_id	coffee_id	order_id	Comments	StarRating
101	1	C1	1	Great coffee, loved the flavor!	5
102	2	C2	2	Amazing cappuccino, very creamy.	4.5
103	3	C3	3	Excellent latte, will order again.	5
104	4	C4	4	Not satisfied with the black coffee.	2
105	5	C5	5	Delicious macchiato, perfect balance.	4
NULL	NULL	NULL	NULL	NULL	NULL

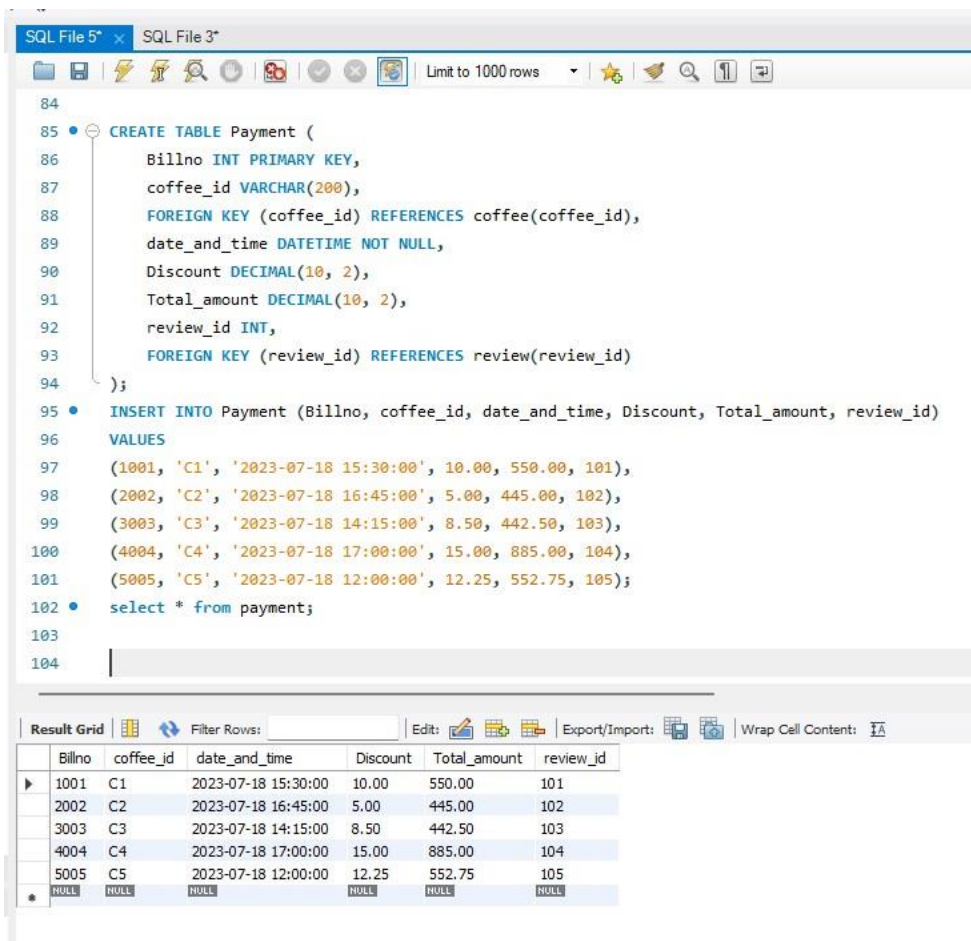
```

CREATE TABLE Payment (Billno INT PRIMARY KEY, coffee_id VARCHAR(200),
    FOREIGN KEY (coffee_id) REFERENCES coffee(coffee_id),
    date_and_time DATETIME NOT NULL, Discount DECIMAL(10, 2), Total_amount DECIMAL(10, 2),
    review_id INT,
    FOREIGN KEY (review_id) REFERENCES review(review_id)
);

INSERT INTO Payment (Billno, coffee_id, date_and_time, Discount, Total_amount, review_id)
VALUES
(1001, 'C1', '2023-07-18 15:30:00', 10.00, 550.00, 101),
(2002, 'C2', '2023-07-18 16:45:00', 5.00, 445.00, 102),
(3003, 'C3', '2023-07-18 14:15:00', 8.50, 442.50, 103),
(4004, 'C4', '2023-07-18 17:00:00', 15.00, 885.00, 104),
(5005, 'C5', '2023-07-18 12:00:00', 12.25, 552.75, 105);

select * from payment;

```



The screenshot shows a SQL IDE window with a toolbar and a code editor. The code editor contains the SQL commands from the previous block. Below the code editor is a 'Result Grid' tab showing the output of the 'select * from payment;' query. The grid has 7 columns: Billno, coffee_id, date_and_time, Discount, Total_amount, and review_id. It contains 5 rows of data corresponding to the inserted records.

Billno	coffee_id	date_and_time	Discount	Total_amount	review_id
1001	C1	2023-07-18 15:30:00	10.00	550.00	101
2002	C2	2023-07-18 16:45:00	5.00	445.00	102
3003	C3	2023-07-18 14:15:00	8.50	442.50	103
4004	C4	2023-07-18 17:00:00	15.00	885.00	104
5005	C5	2023-07-18 12:00:00	12.25	552.75	105