Date:

Task:12

USE CASE :- Gift Coupon Application

Aim: A Gift Coupon Application handles offers and payment-related information for customers. Since financial transactions and coupon redemption involve sensitive and real-time data, the system requires strong consistency, reliability and scalability.

Database system - SQL

- ACID Compliance:-

Ensures that every transaction is processed reliably.

Ex: If payment fails, the coupon status remains unchanged.

- Immediate Consistency:-

Unlike No SQL databases, SQL databases maintain strong consistency after every transaction.

- Horizontal Scalability:-

Through SQL clustering, the database can handle high volumes of read and write operations efficiently.

- Transaction Control:-

SQL supports commands like commit, ROLLBACK and SAVE POINT to manage and recover transactions safely.

Transaction processing and Dead lock Control:-

Property	Description	Example
Atomicity	A transaction completes fully (or) not at all	Payment & coupon redemption occur together
Consistency	Ensures data integrity before and after the transaction	Coupon balance remains valid
Isolation	Prevents transactions from interfering	Two users redeeming coupons don't affect each other.
Durability	Data is permanent once Committed	Transaction records persist even after a crash.

Data base Design and Normalization:-

① Users

```
CREATE TABLE users (user ID INT, PRIMARY KEY, User Name VARCHAR(100)
Email VARCHAR(100), Phone VARCHAR(15));
```

② Coupons

```
CREATE TABLE Coupons ((Coupon ID INT PRIMARY KEY, Coupon code VARCHAR(50),
Value DECIMAL (10,2), Expiry Date DATE);
```

③ Transactions

```
CREATE TABLE Transactions (TxnID INT PRIMARY KEY, UserID
INT, Coupon ID INT, Amount DECIMAL (10,2), Txn Date DATETIME,
status VARCHAR (20), FOREIGN KEY (User ID) REFERENCES USERS
(User ID), FOREIGN KEY (Coupon ID) REFERENCES Coupons
(CouponID));
```

④ Payments

```
CREATE TABLE payments (Payment ID INT PRIMARY KEY, Txn IDINT
Method VARCHAR(50), Payment status VARCHAR(50), Payment status
VARCHAR (50), Payment Date DATE TIME, FOREIGN KEY(TxnID)
REFERENCES Transactions (TxnID));
```

Input:-

```
Insert Into Users Values (1, 'Ravi Kumar', 'ravi@gmail.com'), (2, 'SnehaPatel',
'sneha@gmail-com');

Insert Into Coupons VALUES (101, 'NEW100', 100.00, '2025-12-31'),
(102, 'save 50', 50.00, '2025-11-30');
```

Let assume userID = 1(Ravi) uses couponID = 101 (NEW100) on a ₹500 purchase.

```
--- Insert Transaction Record

START TRANSACTION;
Insert Into transactions(Txn ID, UserID, CouponID, Amount, Txn Date,
status) VALUES (1001, 1, 101, 500.00, NOW (), 'PENDING');
--- Record payment information
INSERT Into payments (Payment ID, Txn ID, Method, Payment status,
Payment Date) VALUES (501, 1001, 'UPI', 'success', Now());
--- Update transaction status to 'completed'
Update transactions set status = 'Completed' where TxnID = 1001;
Commit;
```

Users Table

User ID	User Name	Email
1	Ravi kumar	ravi@gmail.com
2	sneha patel	sneha @gmail.com

Coupons Table

Coupon ID	Coupon code	Value	Expiry date
101	NEW100	100.00	2025-12-31
102	SAVE50	50.00	2025-11-30

Transactions Table

Txn ID	User ID	Coupon ID	Amount	Txn Date	Status
1001	1	101	500.00	2025-10-25 16:20:00	Completed

Payments Table

Payment ID	Txn ID	Method	Payment Status	Payment Date
501	1001	UPI	SUCCESS	2025-10-25 16:20:00

Error Handling:

If Payment fails.

```
Start Transaction;
Insert Into transactions (Txn ID, User ID, Coupon ID, Amount, Txn Date,
status) VALUES (1002, 2, 102, 300,00, NOW(), 'PENDING');
Insert Into payments (Payment ID, Txn ID, method, Payments status,
Payment Date) VALUES (502, 1002, 'CARD', 'FAILED', Now ());
Roll back;
```

Result:

All tables are normalized up to BCNF, reducing redundancy and maintaining relational integrity.