Exp.No: 2 **FOREIGN KEY CONSTRAINTS**
Date :

AIM:

To Create a set of tables, add foreign key constraints and incorporate referential integrity.

DESCRIPTION:

SQL FOREIGN KEY Constraint

The FOREIGN KEY constraint is used to prevent actions that would destroy links between tables. FOREIGN KEY is a field (or collection of fields) in one table, that refers to the PRIMARY KEY in another table. The table with the foreign key is called the child table, and the table with the primary key is called the referenced or parent table. The FOREIGN KEY constraint prevents invalid data from being inserted into the foreign key column, because it has to be one of the values contained in the parent table.

SQL FOREIGN KEY on CREATE TABLE

```
CREATE TABLE childTable (col1 int NOT NULL,
col2 int NOT NULL,
col3 int,
………...
PRIMARY KEY (col1),
FOREIGN KEY (col3) REFERENCES parentTable(parent_Primary_key));
```

To allow naming of a FOREIGN KEY constraint, and for defining a FOREIGN KEY constraint on multiple columns, use the following SQL syntax:

```
CREATE TABLE Orders (    OrderID int NOT NULL,     OrderNumber int NOT NULL,     PersonID
        int,     PRIMARY KEY (OrderID),     CONSTRAINT FK_PersonOrder FOREIGN KEY (PersonID)
        REFERENCES Persons(PersonID));
```

SQL FOREIGN KEY on ALTER TABLE

To create a FOREIGN KEY constraint on the "PersonID" column when the "Orders" table is already created, use the following SQL:

```
ALTER TABLE Orders ADD FOREIGN KEY (PersonID) REFERENCES Persons(PersonID);
```

To allow naming of a FOREIGN KEY constraint, and for defining a FOREIGN KEY constraint on multiple columns, use the following SQL syntax:

```
ALTER TABLE Orders ADD CONSTRAINT FK_PersonOrder
     FOREIGN KEY (PersonID) REFERENCES Persons(PersonID);
```

DROP a FOREIGN KEY Constraint

To drop a FOREIGN KEY constraint, use the following SQL:

```
ALTER TABLE Orders DROP CONSTRAINT FK_PersonOrder;
```

QUESTIONS

1. **Write a query to create table which doesn't have a foreign key.**

```
CREATE TABLE Customers (id INT,   first_name VARCHAR(40),   last_name VARCHAR(40),   age INT,
  country VARCHAR(10),   CONSTRAINT CustomersPK PRIMARY KEY (id));
```

2**. Write a query to Adding foreign key to the customer_id field.**

```
CREATE TABLE Orders (   order_id INT,   item VARCHAR(40),   amount INT,
  customer_id INT REFERENCES Customers(id),   CONSTRAINT OrdersPK PRIMARY KEY
(order_id));
```

3. Write a query to Inserting record in table with no foreign key first

```
INSERT INTO Customers VALUES (1, 'John', 'Doe', 31, 'USA'), (2, 'Robert', 'Luna', 22, 'USA');

INSERT INTO Orders VALUES (1, 'Keyboard', 400, 2), (2, 'Mouse', 300, 2), (3, 'Monitor', 12000, 1);
```

4. **Write a query to Adding foreign key to the customer_id field and the foreign key references to the id field of the Customers table**

```
CREATE TABLE Customers1 (id INT,   first_name VARCHAR(40),   last_name VARCHAR(40),   age
INT,
  country VARCHAR(10),   CONSTRAINT CustomersPK PRIMARY KEY (id));

CREATE TABLE Orders1 (order_id INT,   item VARCHAR(40),   amount INT,   customer_id INT,
  CONSTRAINT OrdersPK PRIMARY KEY (order_id));

ALTER TABLE Orders ADD FOREIGN KEY (customer_id) REFERENCES Customers1(id);
```

5. **Write a query to Adding foreign key to the buyer and seller field**

```
CREATE TABLE Users (   id INT,   first_name VARCHAR(40),   last_name VARCHAR(40),   age INT,
  country VARCHAR(10),   CONSTRAINT CustomersPK PRIMARY KEY (id) );


CREATE TABLE Transactions (  transaction_id INT,  amount INT,   seller INT REFERENCES Users(id),
  buyer INT REFERENCES Users(id),   CONSTRAINT TransactionsPK PRIMARY KEY
(transaction_id));
```

6. **How do you create a self-referencing foreign key?"**

```
CREATE TABLE OurStuff( StuffID INT NOT NULL PRIMARY KEY,   StuffSubID INT NULL,
  StuffName VARCHAR(10) NOT NULL,   CONSTRAINT fk_StuffID FOREIGN KEY (StuffSubID)
    REFERENCES OurStuff(StuffID) );

   INSERT INTO OurStuff VALUES
   (1001, NULL, 'stuff1'),
   (1002, 1001, 'stuff2'),
   (1003, 1002, 'stuff3'),
   (1004, 1002, 'stuff4'),
   (1005, 1003, 'stuff5');
```

7. **How do I view all the primary keys and foreign keys in my database?”**

If all you’re after is a basic list of primary key and foreign key constraints, you can use the **information_schema.table_constraints** system view to retrieve the information you need. Let’s start with a couple tables.

```
CREATE TABLE StuffType(TypeID INT NOT NULL, AltID INT NOT NULL, TypeName VARCHAR(10) NOT NULL,
CONSTRAINT pk_TypeID PRIMARY KEY (TypeID), CONSTRAINT uq_AltID UNIQUE (AltID));

CREATE TABLE OurStuff(StuffID INT NOT NULL, StuffName VARCHAR(10) NOT NULL, OurTypeID INT NULL,
OurAltID INT NULL, CONSTRAINT pk_StuffID PRIMARY KEY (StuffID), CONSTRAINT fk_StuffType1 FOREIGN
KEY (OurTypeID) REFERENCES StuffType(TypeID), CONSTRAINT fk_StuffType2 FOREIGN KEY (OurAltID)
REFERENCES StuffType(AltID) );
```

```
SELECT table_name AS TableName, constraint_name AS ConstraintName, constraint_type AS ConstraintType
FROM information_schema.table_constraints ORDER BY TableName;
```

8. Write a query to create a FOREIGN KEY constraint on the "Accno" column when the"Loan” table which is already created.

```
SQL> alter table Loan add foreign key(Accno) references BankAccount(Accno);
```

VIVA Questions

1. “What are the differences between primary keys and foreign keys in SQL Server?”
2. “How do I view information about a foreign key?”
3. “Does my foreign key have to reference a primary key in another table?”
4. “Should I create an index on a foreign key column?”
5. “How do I view all the primary keys and foreign keys in my database?”

RESULT

Thus the database was created and the SQL queries are written to retrieve information from the database.