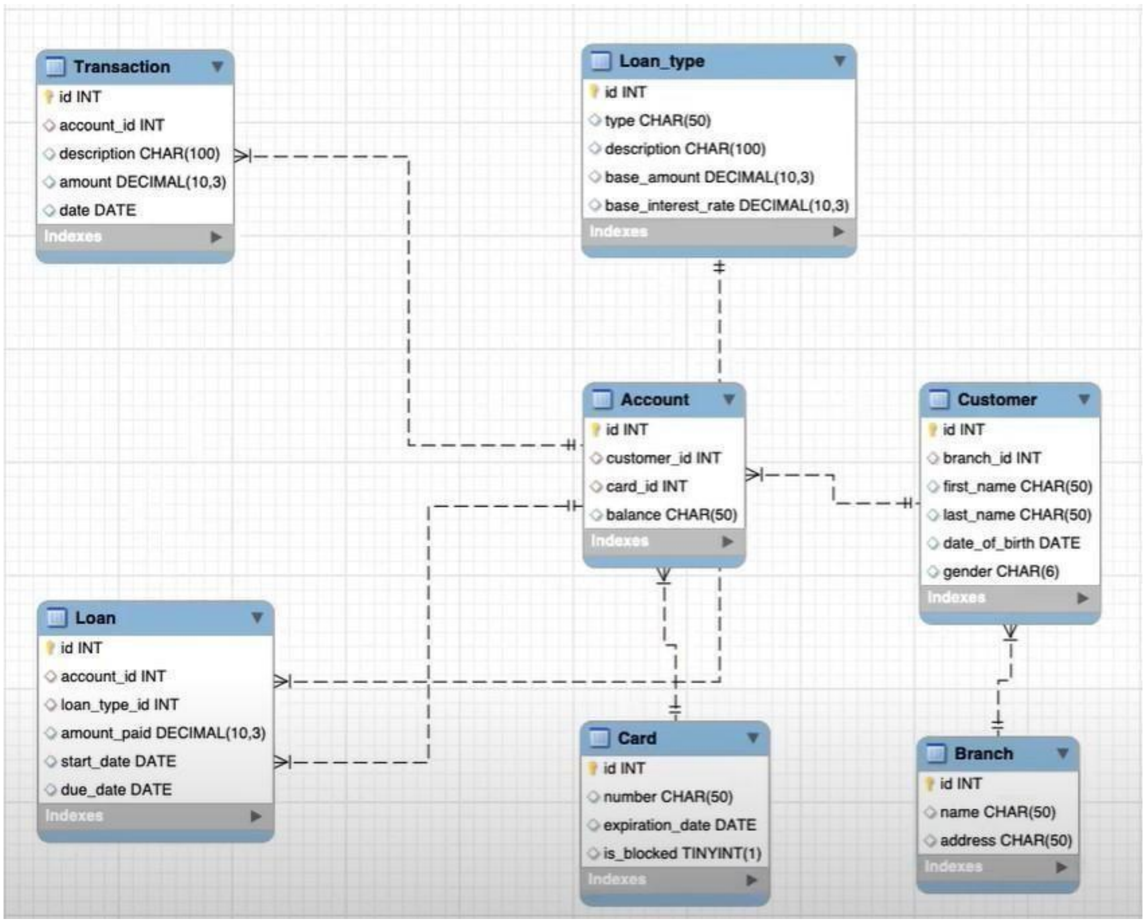
Filtering and Sorting Data

- **Filtering**:

SQL filters are used to obtain a specific subset of the data items from the database. The filter is an SQL WHERE clause that provides a set of comparisons that must be true in order for a data item to be returned.

NOTE - MySQL is case sensitive regarding the data which is present in the table, but not for the general syntax, use of keywords, table name and column name.

Ex-



For the above ER Diagram, To get data associated with a particular address we will use a filter. Given below, gives us a branch that has Delhi as an address.

SELECT *
FROM Branch
WHERE address = “Delhi”;

Here, address is the key and “Delhi” is the value which makes the key-value pair for the WHERE clause condition.

Ex- For the same ER Model - We want to find a loan with an interest rate of either 25% or 35%.

```
SELECT *
FROM Loan_type
WHERE base_interest_rate = 25 OR base_interest_rate = 35;
```

In the above examples, we saw how we can use filters i.e., conditionals inside the WHERE clause.

Ex - For the same ER Model - We want to find customer details whose date of birth is before 16th June 2000.

```
SELECT *
FROM Customer
WHERE date_of_birth < ’16-06-2000’;
```

General Form -

The general form of the **WHERE** clause in our query as a Conditional statement to filter the data is given below:

```
SELECT column_name(s)
FROM T_name
WHERE conditions;
```

- **BETWEEN Operator** –

The BETWEEN operator in WHERE clause selects value within the specified range, so that we do not need to specify all entries exhaustively.

Syntax:

```
SELECT <column-name>
FROM <table-name>
WHERE <column-name>
BETWEEN value-1 AND value-2;
```

To filtering the strings:

- **Wildcards:**

Symbol	Description
%	Represents zero or more characters
_	Represents a single character

Few examples:

1. 'a%' - Find any value that starts with "a".
2. '%or%' - Finds any values that have "or" in any position.
3. '_r%'- Finds any values that have "r" in the second position.
4. 'a%o'- Finds any values that start with "a" and end with "o"

Ex- Write a SQL query to fetch branch id whose address starts with B,

```
SELECT id
FROM Branch
WHERE address = “B%”;
```

We use wildcards with LIKE operators.

Query: -

```
SELECT column_name(s)
FROM T_name
WHERE column_name
LIKE '%o_r%';
```

- **Filtering Operators**

1. **IN operator:** The IN operator allows specifying multiple values in a WHERE clause. Note: - We can use multiple OR operators in place of IN operators as well.

   General Form:

   ```
   SELECT column_name(s)
   FROM table_name
   WHERE column_name IN (value1, value2, ...);
   ```

   Equivalent-

   ```
   SELECT column_name(s)
   FROM table_name
   WHERE column_name = value1 OR column_name = value2 OR column_name = value3,
   .......;
   ```

2. **NOT IN operator:** NOT IN operator is the same as IN operator in syntax and it just negates the conditional i.e., NOT IN operator will return true for only conditionals which are false for IN operator.

   General Form:

   ```
   SELECT column_name(s)

   FROM table_name

   WHERE column_name NOT IN (value1, value2, ...);
   ```

   The above query will return column_name which is not equal to value1, value2, etc. which is nothing but a negation of what we get from IN operator.

3. **IS NULL:** The IS NULL operator is used to test for empty values (NULL values).

   Ex- We want to find the branch ID for which address is NULL, the query is given below –

   ```
   SELECT branch_id
   FROM branch
   WHERE address IS NULL;
   ```

Similarly, we also have IS NOT NULL operator which we use to check for non-empty values.

Ex: We want to find a branch ID for which address is not NULL, a query is given below-

```
SELECT branch_id
FROM branch
WHERE address IS NOT NULL;
```

Apart from these operators, there are several comparison operators which can be used to filter data. These operators are particularly used when filtering is based on some numeric fields. The various comparison operators are shown in the table below:

Operator	Meaning
=	Equal to
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to
<>	Not equal to

General Form:

```
SELECT columnList(s)
FROM table_name
WHERE column_name operator value(s);
```

There are some Logical operators which can also be used to filter data. Following are some logical operators that can be used to filter data along with the syntax of writing the query with the specific operator.

Operator	Meaning
AND	Returns true if both component conditions are true
OR	Returns true if either component condition is true
NOT	Returns true if the following condition is false

- AND requires both conditions to be true:

```
SELECT columnList(s)
FROM table_name
WHERE condition1 AND column_name LIKE ‘%a_;
```

- OR requires either condition to be true.

```
SELECT columnList(s)
FROM table_name
WHERE condition1 OR column_name LIKE ‘%a_;
```

- NOT operator

```
SELECT columnList(s)
FROM table_name
WHERE column_name NOT IN (value1, value2, ...);
```

Order of precedence is the order in which the filtering conditions will be evaluated. This is one of the important concepts when you are preparing for interviews. The order of precedence of the filtering component of SQL is given below:

Order Evaluated	Operator
1	Arithmetic operators
2	Concatenation operator
3	Comparison conditions
4	IS [NOT] NULL, LIKE, [NOT] IN
5	[NOT] BETWEEN
6	NOT logical condition
7	AND logical condition
8	OR logical condition

- **Sorting**:

  Sorting is simply done using the ORDER BY keyword which is used to sort the result set in ascending or descending order.

  General form:

```
SELECT column1, column2, ...
FROM table_name
ORDER BY column1, column2, ... ASC|DESC;
```

  Ex- Enlist account ID in ascending order.

```
SELECT id
FROM account
ORDER BY id ASC;
```

Q. What is the difference between Sorting and Filtering?

Ans. Sorting is arranging the data in ascending or descending order according to the user's needs

Whereas Filtering is used to fetch the data that a user might need for a use case.