**GROUP NUMBER: 30**

**Customer Complaint Analysis**

**Submitted by:**

* **Akshat Khapra**
* **Saikrishnagattu**
* **Kancharla Manvith**
* **Yalavarthi Akhil Sai**

**Complaint analysis** is one of the important feature to get analyzed when dealing with the customers.

When a customer files a complaint, he does voice a concern in relation to that particular product or the service provided. However, it is not necessary to treat all the complaints equally.

**Abstract:**

There has been the necessity to analyze such data because of the rapid increase in the quantity of customer data. This project provides a method to analyse customer complaints. The analysis identified two needs that experts had not identified but regarded as important. This project helps to identify requirements of all the points at which customers want to obtain help from the product.

**Project Details:**

Here, in our database that we have created for our project, we have created tables as follows:

* Customer\_Data
* Table\_Home
* Work\_Order
* Complaint\_Order
* Order\_Item

Detailed explanation of all the tables created for the required data.

**Queries used:**

1. **Create:** statement is used to create a new table in a database.
2. **Not NULL:** The NOT NULL constraint enforces a column to NOT accept NULL values.
3. **Unique:** The UNIQUE constraint ensures that all values in a column are different.
4. **Primary Key:** The PRIMARY KEY constraint uniquely identifies each record in a table.
5. **Foreign Key:** A FOREIGN KEY is a field (or collection of fields) in one table, that refers to the [PRIMARY KEY](https://www.w3schools.com/sql/sql_primarykey.asp) in another table.
6. **Select:** The SELECT statement is used to select data from a database.
7. **Order By:** The ORDER BY keyword is used to sort the result-set in ascending or descending order.
8. **Insert Into:** The INSERT INTO statement is used to insert new records in a table.

**Customer\_Data** contains all the details of the customer including customer\_id, first\_name, last\_name, DOB, Phone, Email

CREATE TABLE CUSTOMER\_DATA

(

CUSTOMER\_ID\_CUST INT IDENTITY(100, 1) PRIMARY KEY,

FIRST\_NAME\_CUST VARCHAR(20) NOT NULL,

LAST\_NAME\_CUST VARCHAR(20) NOT NULL,

DOB\_CUST DATE NOT NULL DEFAULT CONVERT(DATE, '10/02/2000',10),

PHONE\_CUST CHAR(12) DEFAULT '\*\*\*-\*\*\*-\*\*\*\*',

EMAIL\_ID\_CUST VARCHAR(70) UNIQUE

);

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Customer\_ID | First\_Name | Last\_Name | DOB | Phone | Email |
| 1001 | Kal | wood | 11/2/1997 | 121-123-2367 | [kal.woo@acc.com](mailto:kal.woo@acc.com) |
| 1002 | John | abd | 11/3/1997 | 763-868-8689 | [john.abd@acc.com](mailto:john.abd@acc.com) |
| 1003 | Ben | parl | 11/4/1997 | 726-878-8236 | [ben.par@acc.com](mailto:ben.par@acc.com) |
| 1004 | Mica | rym | 11/5/1997 | 212-121-1231 | [mica.rym@acc.com](mailto:mica.rym@acc.com) |
| 1005 | Tam | rym | 11/6/1997 | 342-768-8768 | [tam.rym@acc.com](mailto:tam.rym@acc.com) |
| 1006 | Mon | hoo | 11/7/1997 | 767-776-7987 | [mon.hoo@acc.com](mailto:mon.hoo@acc.com) |
| 1007 | La | parl | 11/8/1997 | 767-878-8773 | [la.par@acc.com](mailto:la.par@acc.com) |
| 1008 | Del | crup | 11/9/1997 | 727-789-8763 | [del.cru@acc.com](mailto:del.cru@acc.com) |
| 1009 | Yiv | toe | 11/10/1997 | 323-876-6832 | [yiv.toe@acc.com](mailto:yiv.toe@acc.com) |
| 1010 | Alza | lip | 11/11/1997 | 231-878-8768 | [alza.lip@acc.com](mailto:alza.lip@acc.com) |
| 1011 | Dev | chick | 11/12/1997 | 165-767-8682 | [dev.chi@acc.com](mailto:dev.chi@acc.com) |
| 1012 | Neil | mulq | 11/13/1997 | 123-763-8927 | [neil.mul@acc.com](mailto:neil.mul@acc.com) |
| 1013 | Sam | hon | 11/14/1997 | 178-987-6889 | [sam.hon@acc.com](mailto:sam.hon@acc.com) |
| 1014 | Shaw | Valak | 11/15/1997 | 868-079-8698 | [shaw.Val@acc.com](mailto:shaw.Val@acc.com) |
| 1015 | Philip | Gibe | 11/16/1997 | 768-868-5767 | [philip.Gib@acc.com](mailto:philip.Gib@acc.com) |
| 1016 | GR | Brus | 11/17/1997 | 775-886-7657 | [gr.Bru@acc.com](mailto:gr.Bru@acc.com) |
| 1017 | Aaron | Gig | 11/18/1997 | 323-768-8683 | [aaron.Gig@acc.com](mailto:aaron.Gig@acc.com) |
| 1918 | Nik | Ben | 11/19/1997 | 322-786-8682 | [nik.Ben@acc.com](mailto:nik.Ben@acc.com) |
| 1019 | Paul | Den | 11/20/1997 | 126-687-7868 | [paul.Den@acc.com](mailto:paul.Den@acc.com) |
| 1020 | Shag | Poiu | 11/21/1997 | 125-767-7889 | [shag.Poi@acc.com](mailto:shag.Poi@acc.com) |

**Table\_Home** contains all the address details of the customer including Home\_id, address, city and country.

CREATE TABLE TABLE\_HOME

(

HOME\_ID\_HME INT IDENTITY(200, 1),

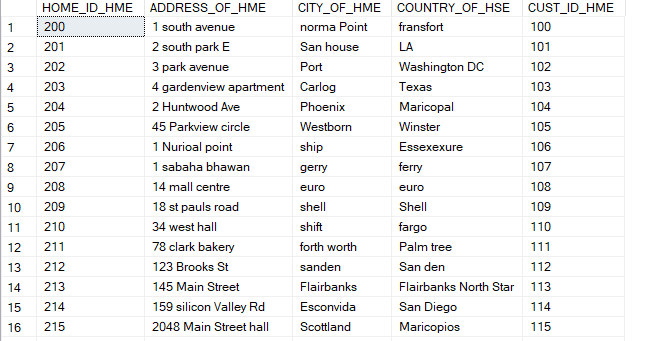
ADDRESS\_OF\_HME VARCHAR(150) NOT NULL,

CITY\_OF\_HME VARCHAR(50) NOT NULL,

COUNTRY\_OF\_HSE VARCHAR(50) NOT NULL,

CUST\_ID\_HME INT NOT NULL REFERENCES CUSTOMER\_DATA (CUSTOMER\_ID\_CUST),

CONSTRAINT PK\_KEY PRIMARY KEY (HOME\_ID\_HME, CUST\_ID\_HME)

) 

**Work\_Order** contains the order id, cust work id, home work id, type of work, created date, completed work date

CREATE TABLE WORK\_ORDER

(

ORDER\_ID\_WORK INT IDENTITY(300, 1) PRIMARY KEY,

CUST\_ID\_WORK INT NOT NULL,

HOME\_ID\_WORK INT NOT NULL,

TYPE\_0F\_WORK VARCHAR(20) NOT NULL,

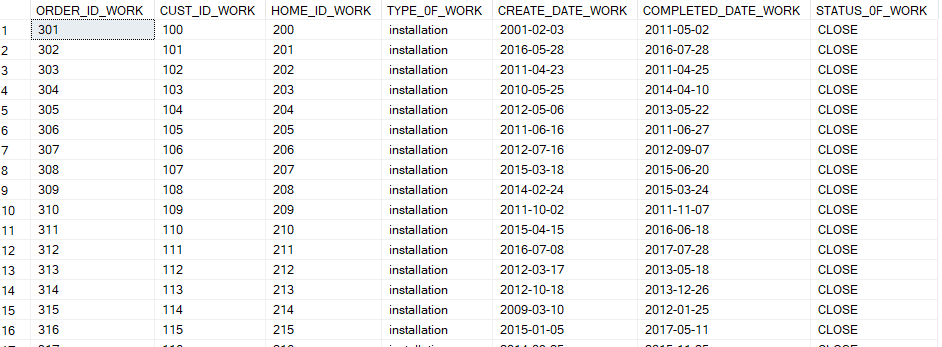
CREATE\_DATE\_WORK DATE NOT NULL,

COMPLETED\_DATE\_WORK DATE NOT NULL DEFAULT CONVERT(DATE, '10/02/2000',201),

STATUS\_0F\_WORK VARCHAR(15) NOT NULL,

CONSTRAINT FK\_KEY\_WORK FOREIGN KEY(HOME\_ID\_WORK, CUST\_ID\_WORK) REFERENCES TABLE\_HOME(HOME\_ID\_HME, CUST\_ID\_HME)

)



**Complaint\_Order** table contains order id complaint, customer id complaint order, home id complaint order, complaint order type, complaint order created date, complaint order status.

CREATE TABLE COMPLAINT\_ORDER

(

ORDER\_ID\_COMPLAINT INT IDENTITY(4000001, 1) PRIMARY KEY,

CUST\_ID\_COMPLAINT\_ORDER INT NOT NULL,

HOME\_ID\_COMPLAINT\_ORDER INT NOT NULL,

TYPE\_COMPLAINT\_ORDER VARCHAR(20) NOT NULL,

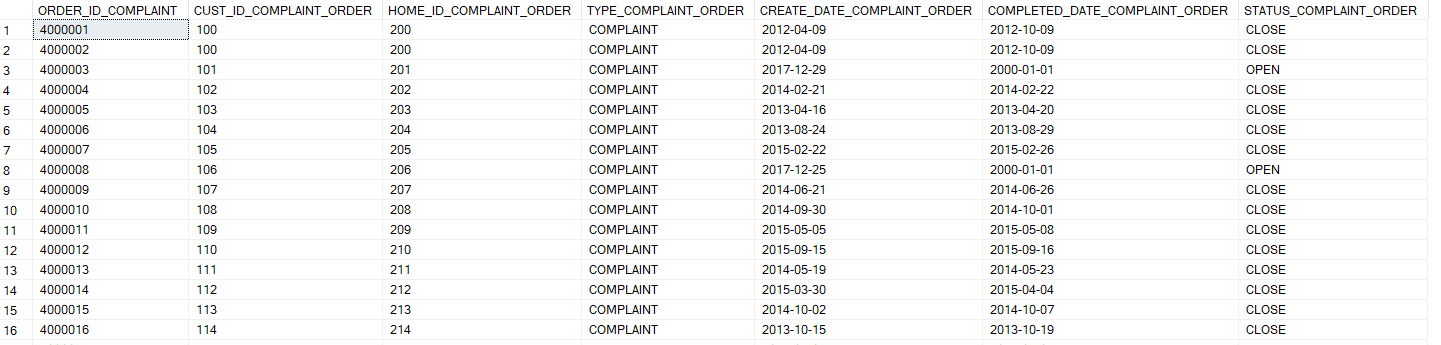
CREATE\_DATE\_COMPLAINT\_ORDER DATE NOT NULL,

COMPLETED\_DATE\_COMPLAINT\_ORDER DATE NOT NULL DEFAULT CONVERT(DATE, '10/02/2000',301),

STATUS\_COMPLAINT\_ORDER VARCHAR(15) NOT NULL,

CONSTRAINT FK\_KEY\_CO FOREIGN KEY(HOME\_ID\_COMPLAINT\_ORDER, CUST\_ID\_COMPLAINT\_ORDER) REFERENCES TABLE\_HOME(HOME\_ID\_HME, CUST\_ID\_HME)

);



**Order\_item** table contains item order id, mode of item, generic\_item, rate of item.

CREATE TABLE ORDER\_ITEM (

ORDER\_ID\_ITEM INT NOT NULL REFERENCES WORK\_ORDER(ORDER\_ID\_WORK),

MODE\_OF\_ITEM VARCHAR (30) NOT NULL,

GENRERIC\_ITEM VARCHAR(20) NOT NULL CHECK(GENRERIC\_ITEM IN ('NEWS', 'MOVIE', 'SPORTS', 'MUSIC')),

RATE\_OF\_ITEM INT NOT NULL

);

|  |  |  |  |
| --- | --- | --- | --- |
| Order\_id | Mode\_of\_item | Generic\_item | rate\_of\_item |
| 101 | NDTV | NEWS | 25 |
| 102 | ZEE | SPORTS | 30 |
| 103 | ESPN | MOVIE | 27 |
| 104 | 9XM | MUSIC | 29 |
| 105 | Star Pix | NEWS | 25 |
| 106 | NDTV | SPORTS | 24 |
| 107 | ZEE | MOVIE | 22 |
| 108 | ESPN | MUSIC | 20 |
| 109 | 9XM | NEWS | 29 |
| 110 | Star Pix | SPORTS | 27 |

**Design :**

We have made five tables and connected each other.

**Software Used:**

We have used microsoft sql server management studio

**Functionality of Database:**

The above created tables, will have all the data related to the customer: personal details of the customer, the address associated to the customer, order item, all the details related to that, complaint details, when it was created and by when it was resolved and all other details associated to that.

Having all the details correctly maintained and analysed helps in dealing with the data correctly and more efficiently, else it is hard to maintain what customer has filed what complaint and by when and how it was resolved.

**Target Audience for the Project created:**

Target audience remains who all will be associated with the product manufactured and sold to. The ones having any sort of complaints will have their data added into this system. Whensoever the product is sold, a customer id is given to the customer, and likewise with that customer\_id user has to file complaint in case any. Similarly other associated data to that customer\_id can be fetch like the personal details and home address of the customer and so on.

Such project can be used to have a proper maintained and feasible data in any field, say we have a school, and parents can have complaints, that can be managed in similar sense, same gores with hospitals, restaurants and other service providing industries.

**Join statement** is used to combine data or rows from two or more tables based on a common field between them. Different types of Joins are:

* INNER JOIN
* LEFT JOIN
* RIGHT JOIN
* FULL JOIN

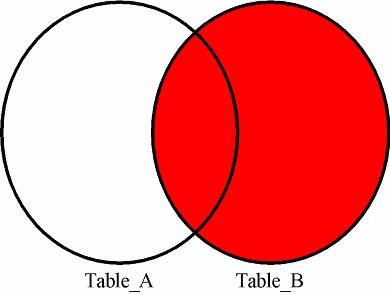
**INNER JOIN:** The INNER JOIN keyword selects all rows from both the tables as long as the condition satisfies. This keyword will create the result-set by combining all rows from both the tables where the condition satisfies i.e value of the common field will be same.



**LEFT JOIN**: This join returns all the rows of the table on the left side of the join and matching rows for the table on the right side of join. The rows for which there is no matching row on right side, the result-set will contain null. LEFT JOIN is also known as LEFT OUTER JOIN



**RIGHT JOIN**: RIGHT JOIN is similar to LEFT JOIN. This join returns all the rows of the table on the right side of the join and matching rows for the table on the left side of join. The rows for which there is no matching row on left side, the result-set will contain null. RIGHT JOIN is also known as RIGHT OUTER JOIN



**FULL JOIN:** FULL JOIN creates the result-set by combining result of both LEFT JOIN and RIGHT JOIN. The result-set will contain all the rows from both the tables. The rows for which there is no matching, the result-set will contain NULL values.



**Similar queries has been implemented in the following examples:**

**Q1: In order to post welcome letters and user guides to customers, dispatch team need**

**customer name, address and contact details. Write SQL query to get desired info.**

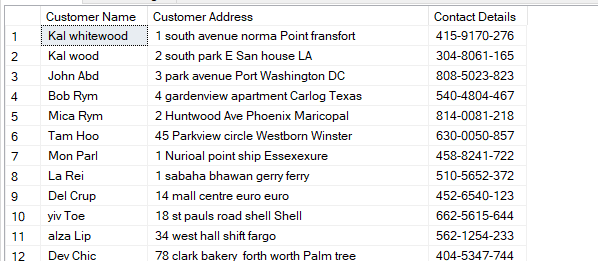
SELECT

CONCAT(First\_name\_cust, SPACE(1), last\_name\_cust) 'Customer Name',

CONCAT(ADDRESS\_OF\_HME, SPACE(1), CITY\_OF\_HME, SPACE(1), COUNTRY\_OF\_HSE) 'Customer Address',

PHONE\_CUST 'Contact Details'

FROM Customer\_data INNER JOIN TABLE\_HOME ON CUSTOMER\_ID\_CUST = CUST\_ID\_HME



**Q2: Get the details of customers who are using our services in one location.**

SELECT

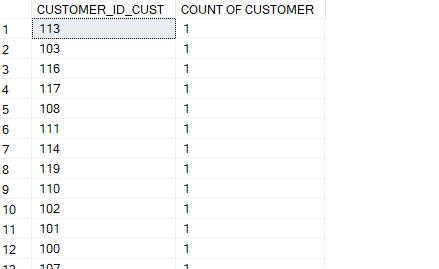
CUSTOMER\_ID\_CUST,

COUNT(CUST\_ID\_HME) AS 'COUNT OF CUSTOMER'

FROM Customer\_data INNER JOIN TABLE\_HOME ON CUSTOMER\_ID\_CUST = CUST\_ID\_HME

GROUP BY CUSTOMER\_ID\_CUST

HAVING COUNT(CUST\_ID\_HME) = 1



**Q3: Which are the customers that has given their house details.**

SELECT

CUSTOMER\_ID\_CUST,

CUST\_ID\_HME

FROM Customer\_data LEFT JOIN TABLE\_HOME ON CUSTOMER\_ID\_CUST = CUST\_ID\_HME

WHERE CUST\_ID\_HME IS NOT NULL

SELECT

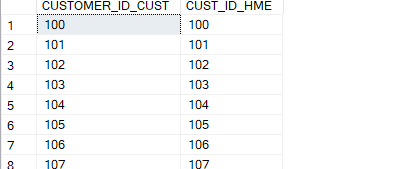
CUSTOMER\_ID\_CUST,

COUNT(CUST\_ID\_HME) as 'Count House Details'

FROM Customer\_data LEFT JOIN TABLE\_HOME ON CUSTOMER\_ID\_CUST = CUST\_ID\_HME

GROUP BY CUSTOMER\_ID\_CUST

HAVING COUNT(CUST\_ID\_HME) != 0



**Q4: Get the install dates corresponding to all customers in different locations.**

SELECT

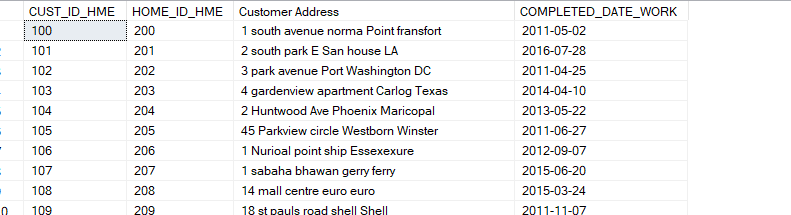
CUST\_ID\_HME,

HOME\_ID\_HME,

CONCAT(ADDRESS\_OF\_HME, SPACE(1), CITY\_OF\_HME, SPACE(1), COUNTRY\_OF\_HSE) 'Customer Address',

COMPLETED\_DATE\_WORK

FROM TABLE\_HOME INNER JOIN WORK\_ORDER ON CUST\_ID\_HME = CUST\_ID\_WORK AND HOME\_ID\_HME = HOME\_ID\_WORK where TYPE\_0F\_WORK = 'installation' and STATUS\_0F\_WORK = 'close'

****

**Q5: Get the customer name and contact details of the customers along with other info extracted in Q4 above.**

SELECT

CONCAT(FIRST\_NAME\_CUST, SPACE(1), LAST\_NAME\_CUST) 'Customer Name',PHONE\_CUST

CUSTOMER\_ID\_HSE,

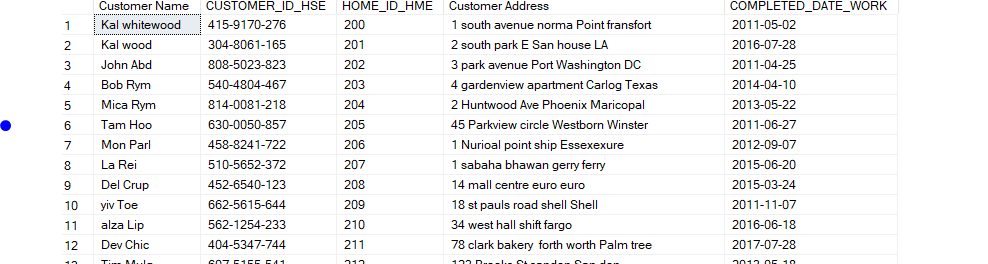
HOME\_ID\_HME,

CONCAT(ADDRESS\_OF\_HME, SPACE(1), CITY\_OF\_HME, SPACE(1), COUNTRY\_OF\_HSE) 'Customer Address',

COMPLETED\_DATE\_WORK

FROM TABLE\_HOME INNER JOIN WORK\_ORDER ON CUST\_ID\_HME = CUST\_ID\_WORK AND HOME\_ID\_HME = HOME\_ID\_WORK

INNER JOIN CUSTOMER\_DATA ON CUSTOMER\_ID\_CUST = CUST\_ID\_HME where TYPE\_0F\_WORK = 'installation' and STATUS\_0F\_WORK = 'close'



**References :**

<https://www.w3schools.com/sql/sql_insert.asp>

<https://www.geeksforgeeks.org/sql-tutorial/>

**Video link:**

[link: https://youtube.com/watch?v=OAbK93tb3PI&feature=share](link:%20https://youtube.com/watch?v=OAbK93tb3PI&feature=share)