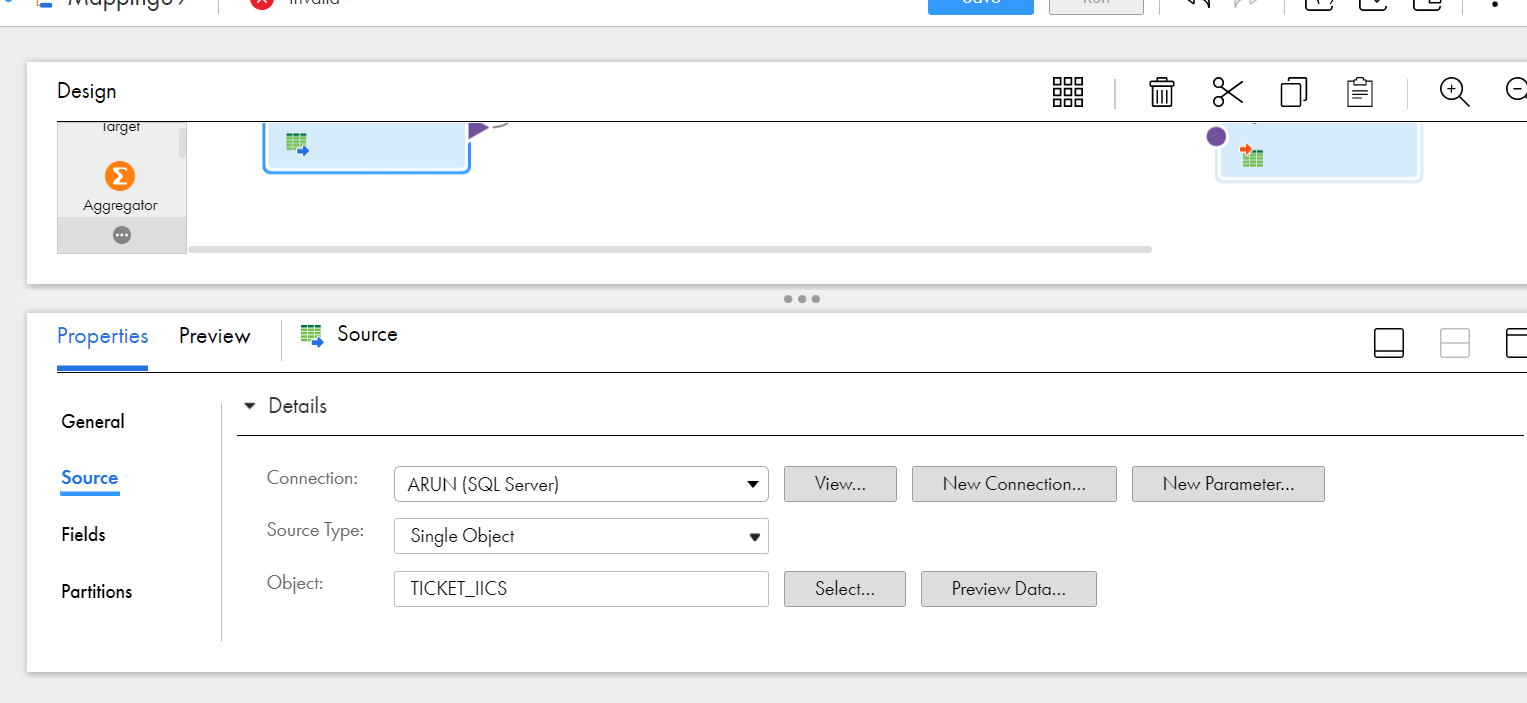
IICS

SCD1

SELECTING THE SOURCE



ADDING ALL THE INPUT COLUMNS AS OUTPUT COLUMNS TO PERFORM THE CHECKSUM OPERATION

Graphical user interface, text, application, email

Description automatically generated

PERFORMING THE CHECKSUM OPERATION USING MD5 METHOD

Graphical user interface, application

Description automatically generated

NEXT STEP WE NEED TO SELECT THE LOOK UP OBJECT AND VALIDATE THE LOOKUP CONDTION

Graphical user interface, text, application

Description automatically generated

PERFORMING EXPRESSION TASK TO INSERT AND UPDATE FUNCTION

Graphical user interface, application

Description automatically generated

ROUTER OPERATION TO GET THE INSERT AND UPDATE OUTPUTS

Graphical user interface, diagram, application

Description automatically generated

INSERT EXPRESSION TASK

Graphical user interface, text, application, email

Description automatically generated

UPDATE EXPRESSION TASK

Graphical user interface, application

Description automatically generated

SELECTING TWO TARGETS AND PERFORMING THE FIELD MAPPING AND UPDATING THE COLUMN WHICH NEEDS TO BE UPDATED

TARGET 1 FIELD MAPPINGGraphical user interface, text, application, email

Description automatically generated

TARGET 2 FIELD MAPPINGGraphical user interface, text, application, email

Description automatically generated

AFTER VALID MAPPING WE ARE RUNNING THE TASK

Graphical user interface, application

Description automatically generated

RESULTS AFTER SUCCEFULL RUN IN DB

THE DATA IS INSERTED

Table

Description automatically generated

NOW UPDATING THE ROW AND CHECHKING THE MAPPING

Graphical user interface, text, application

Description automatically generated

SOURCE IS UPDATED

SINCE WE HAVE UPDATED ONLY ONE ROW IT PASSES THE UPDATE ROW ALONE

Graphical user interface, text, application, email

Description automatically generated

UPDATED RESULT

Graphical user interface

Description automatically generated with low confidence

SCD2 IN IICS

SELECTING THE SOURCE

Graphical user interface, text, application

Description automatically generated

ADDING ALL THE INPUT COLUMNS AS OUTPUT COLUMNS TO PERFORM THE CHECKSUM OPERATION

Graphical user interface, text, application, email

Description automatically generated

NEXT STEP WE NEED TO SELECT THE LOOK UP OBJECT AND VALIDATE THE LOOKUP CONDTION Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

FLAGGING OPERATION

Graphical user interface, text, application, email

Description automatically generated

ADDING ROUTER TO 3 OUTPUT FIELDS INSERT,INSCLS,UPDATE

Graphical user interface, application

Description automatically generated

EXPRESSION 1 INSERT

Graphical user interface, text, application, email

Description automatically generated

EXPRESSION 2 INSERT CLOSE

Graphical user interface, text, application, email

Description automatically generated

EXPRESSION 3

Graphical user interface, text, application, email

Description automatically generated

FIELD MAPPING OF ALL 3 TARGETS

Graphical user interface, text, application, email

Description automatically generated

Table

Description automatically generated with medium confidence

Graphical user interface, application

Description automatically generated

RUNNING THE TASK AFTER VALIDATION

Graphical user interface, application

Description automatically generatedGraphical user interface, text

Description automatically generated

DESTINATION RESULT

Table

Description automatically generated

UPDATING THE ROW AND CHECHKING THE UPDATION TASK

2 ROWS ARE UPDATED AND TASK IS SUCCESSFUL IT IS LOADED IS DB ALSO

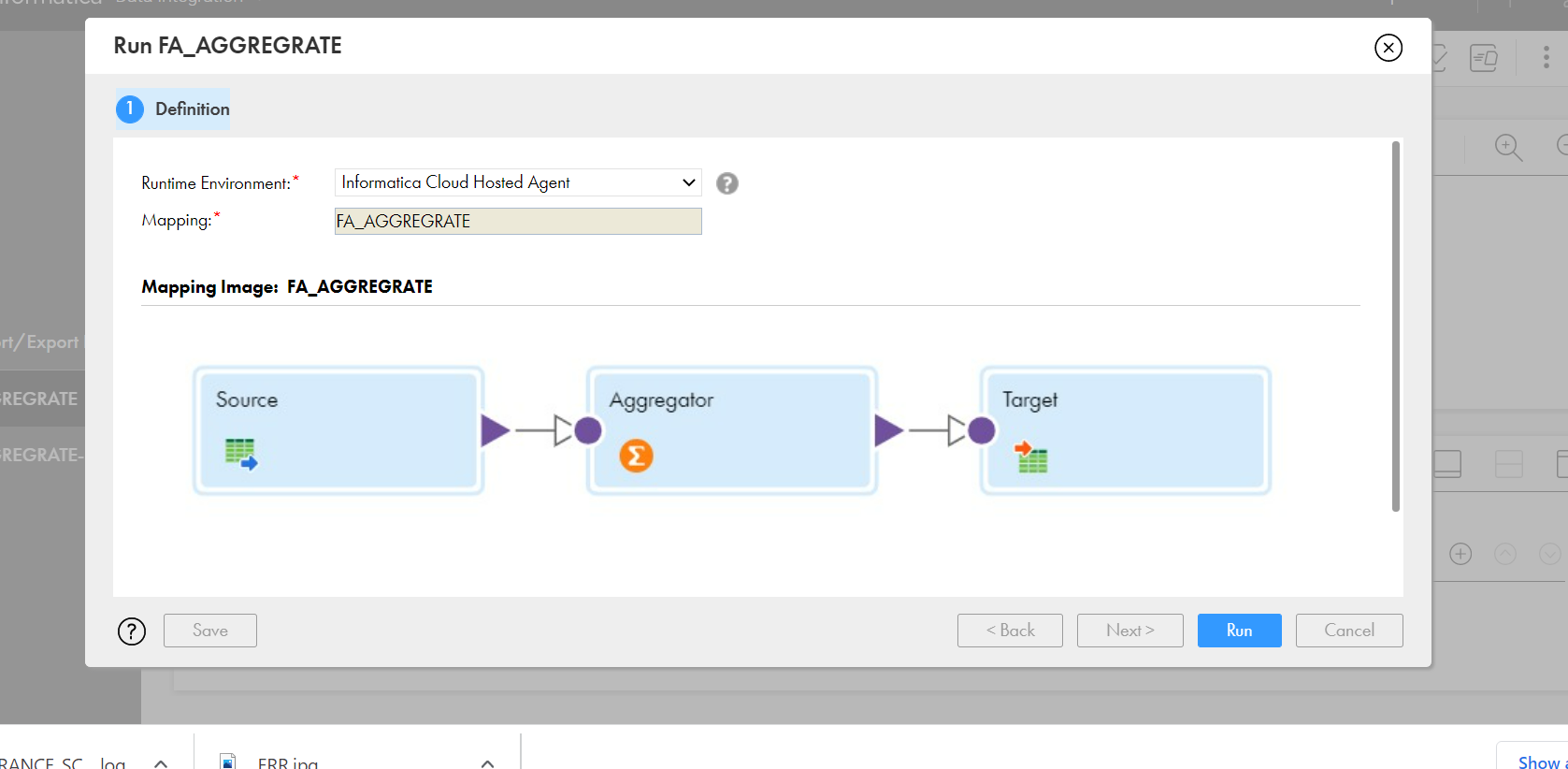
A screenshot of a computer

Description automatically generatedGraphical user interface, text, application, email

Description automatically generated

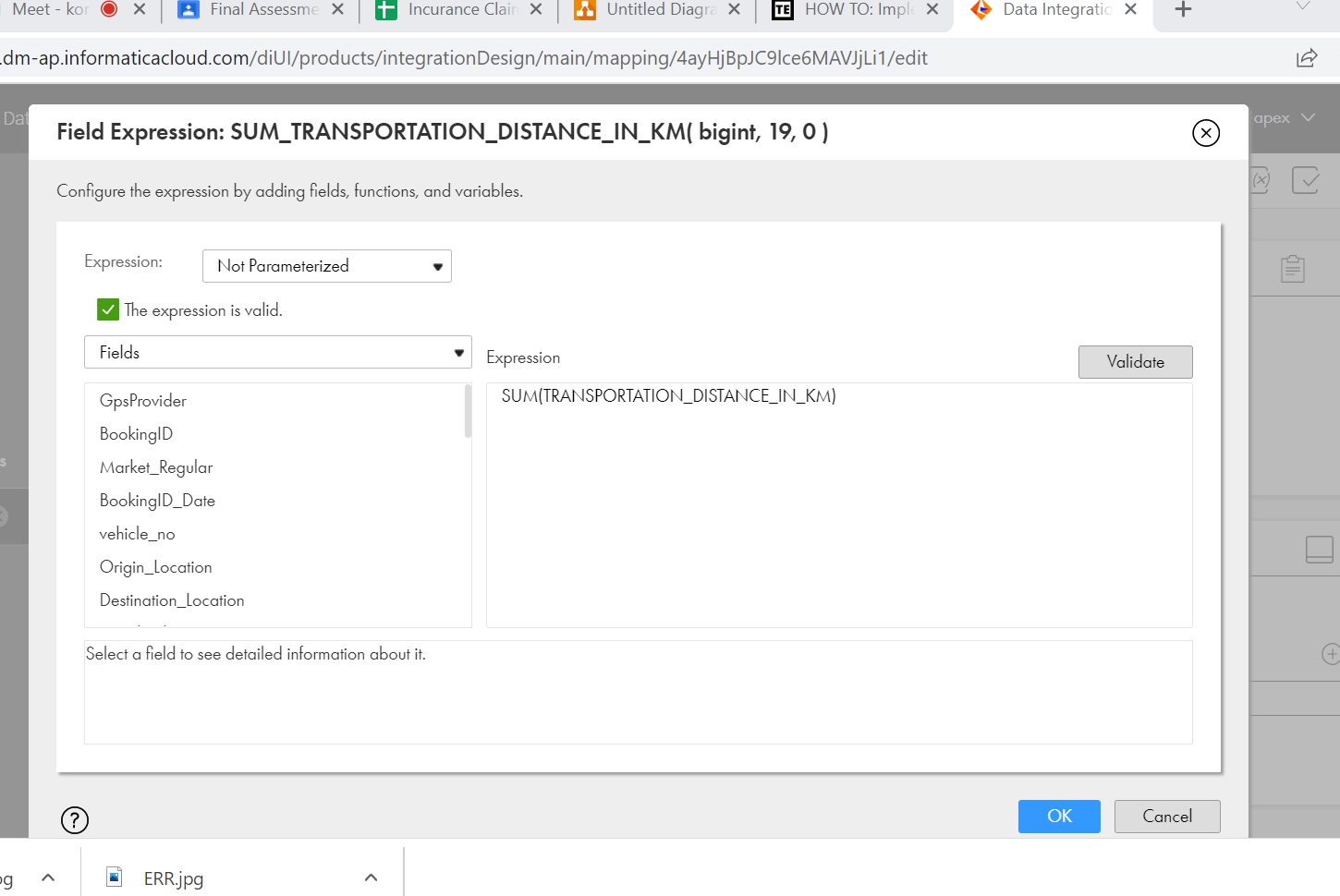
2 ROWS ARE UPDATED AND TASK IS SUCCESSFUL IT IS LOADED IS DB ALSO

AGGREGRATE TRANSFORMATION



MAPPING OG AGGREGRATE TASK

SUM BY AGGREGRATE



AVG AGGREGRATE

Graphical user interface, text, application

Description automatically generatedGraphical user interface, text, application, Word

Description automatically generated

COUNT BY AGGREGRATE

Graphical user interface, application

Description automatically generatedGraphical user interface

Description automatically generated

MIN AGGREGRATE

Graphical user interface, application

Description automatically generatedGraphical user interface, text, application, email

Description automatically generated

MAX AGGREGRATE

Graphical user interface, text, application

Description automatically generatedGraphical user interface, application

Description automatically generated

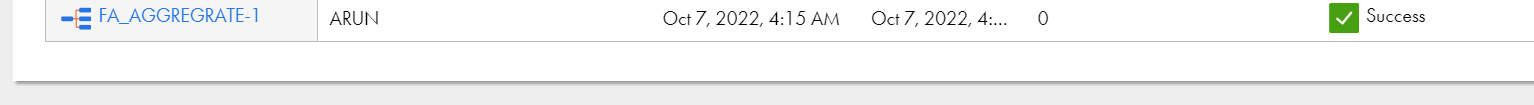
GROUP BY

Graphical user interface, application

Description automatically generated

SUCCESS RESULTS OF AGGRGRATE

Graphical user interface, text, application, email

Description automatically generated

SOURCE TO DESTINATION TRANSFORMATION

TRANSFORMATION FUNCTION USING RANKING

MAPPING OF RANKING

Diagram

Description automatically generated

SUCCESS RESULT OF RANK TRANSFORMATION

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

DATA TRANSFERRED TO DESTINATIONGraphical user interface, application

Description automatically generated

REPLICATION TASK

THIS TASK IS TO REPLICATE SAME TABLE TO OTHER

Graphical user interface, text, application, email

Description automatically generated

SELECTING THE OBJECT TO REPLICATE

Graphical user interface, application, Teams

Description automatically generated

SELECT THE COLUMNS AND OTHER FILTER DETAILS

Graphical user interface, application

Description automatically generatedGraphical user interface

Description automatically generated

RUNNING THE TASK

Graphical user interface, text, application, Teams

Description automatically generated

SUCCESS OF THE TASK

Graphical user interface, text, application, email

Description automatically generated

FINAL RESULT IN THE DB

Graphical user interface, text

Description automatically generatedTable

Description automatically generated

**-----CREATION OF SOURCE TABLES USING GIVEN DATA SOURCE**

**CREATE** **TABLE** FA\_SALES\_2 (

SupplierID **INT**,

SupplierAddress **VARCHAR**(100),

SupplierName **VARCHAR**(100),

SupplierContactDetails **VARCHAR**(100),

ProductID **FLOAT**,

CarMaker **VARCHAR**(100),

CarModel **VARCHAR**(100),

CarColor **VARCHAR**(100),

CarModelYear **INT**,

CarPrice **FLOAT**,

CustomerID **VARCHAR**(100),

CustomerName **VARCHAR**(100),

Gender **VARCHAR**(100),

JobTitle **VARCHAR**(100),

PhoneNumber **VARCHAR**(100),

EmailAddress **VARCHAR**(100),

City **VARCHAR**(100),

Country **VARCHAR**(100),

CountryCode **VARCHAR**(100),

State **VARCHAR**(100),

CustomerAddress **VARCHAR**(100),

OrderDate **INT**,

OrderID **FLOAT**,

ShipDate **INT**,

ShipMode **VARCHAR**(100),

Shipping **VARCHAR**(100),

PostalCode **INT**,

Sales **FLOAT**,

Quantity **INT**,

Discount **VARCHAR**(100),

CreditCardType **VARCHAR**(100),

CreditCard **VARCHAR**(100),

CustomerFeedback **VARCHAR**(100)

)

----DDL OPERATION OF SALES

---CREATE OPERATION--

**CREATE** **TABLE** FA\_SALES\_2 (

SupplierID **INT**,

SupplierAddress **VARCHAR**(100),

SupplierName **VARCHAR**(100),

SupplierContactDetails **VARCHAR**(100),

ProductID **FLOAT**,

CarMaker **VARCHAR**(100),

CarModel **VARCHAR**(100),

CarColor **VARCHAR**(100),

CarModelYear **INT**,

CarPrice **FLOAT**,

CustomerID **VARCHAR**(100),

CustomerName **VARCHAR**(100),

Gender **VARCHAR**(100),

JobTitle **VARCHAR**(100),

PhoneNumber **VARCHAR**(100),

EmailAddress **VARCHAR**(100),

City **VARCHAR**(100),

Country **VARCHAR**(100),

CountryCode **VARCHAR**(100),

State **VARCHAR**(100),

CustomerAddress **VARCHAR**(100),

OrderDate **INT**,

OrderID **FLOAT**,

ShipDate **INT**,

ShipMode **VARCHAR**(100),

Shipping **VARCHAR**(100),

PostalCode **INT**,

Sales **FLOAT**,

Quantity **INT**,

Discount **VARCHAR**(100),

CreditCardType **VARCHAR**(100),

CreditCard **VARCHAR**(100),

CustomerFeedback **VARCHAR**(100)

)

**SELECT** \* **FROM** FA\_SALES\_2

**select** \* **from** FA\_INSURANCE

---DML OPERATION

**INSERT** **INTO** FA\_SALES\_2 **VALUES** (559,'9087 Bonner dro','Trupe',471-08-6486,8876.0,'Infiniti','Q','Orange',2001,757878.29,63187-087,'Eveleen Rittelmeyer','Female' ,'Professor',602-153-9389,'erittelmeyerfd@hao123.com','Phoenix','United States','US','Arizona','1 Independence Circle',43483.0,54868-4381,43624.0,'First Class','Truck',21265.0,779769.27,1,0.58,'jcb',3553500000000000,'Good')

--ddl alter operation

**alter** **table** FA\_SALES\_2 **alter** **column** customerfeedback **varchar**(200)---increasing the datatype length

--update operation

**update** FA\_SALES\_2 **set** Suppliername ='arun' **where** supplierid=1

**select**\***from** FA\_INSURANCE

**select**\***from** FA\_INSURANCE **where** CATEGORY\_NAME = 'NEPHROLOGY' ;--SELECTING PARTICULAR DEPARTMENT

**select**\***from** FA\_INSURANCE **where** CLAIM\_AMOUNT>=29000;---CHECKING CLAIM AMOUNT GRETER THAN OR EQUAL TO 29000

**select**\***from** FA\_INSURANCE **where** CLAIM\_AMOUNT<=9000;---CHECKING CLAIM AMOUNT GRETER THAN OR EQUAL TO 9000

**select**\***from** FA\_INSURANCE **where** CLAIM\_AMOUNT!=9000;--CHECKING CLAIM AMOUNT NOT EQUAL TO 9000

**select**\***from** FA\_INSURANCE **where** CLAIM\_AMOUN<>12008; --NOT EQUAL TO OPERATION

----NULL OPERATIONS

**select**\***from** FA\_INSURANCE **where** CATEGORY\_CODE **IS** **NULL**

**select**\***from** FA\_INSURANCE **where** CATEGORY\_CODE **IS** **NOT** **NULL**

**select**\***from** FA\_INSURANCE **where** CATEGORY\_CODE **IS** **NULL** **OR** AGE **IS** **NULL**

**select**\***from** FA\_INSURANCE **where** AGE **IS** **NULL** **OR** CATEGORY\_CODE **IS** **NOT** **NULL**

**select**\***from** FA\_INSURANCE **where** CASTE\_NAME **IS** **NULL** **AND** CATEGORY\_NAME **IS** **NULL**

-----IN AND NOT IN OPERATIONS

**select**\***from** FA\_INSURANCE **where** CATEGORY\_CODE **IN**('M6','M5')

**select**\***from** FA\_INSURANCE **where** CATEGORY\_CODE **NOT** **IN**(101,201,130,140)

----LIKE OPERATION

**select**\***from** FA\_INSURANCE **where** HOSP\_DISTRICT **LIKE** '%A%'**AND** MANDAL\_NAME **LIKE** 'V%'

**select**\***from** FA\_INSURANCE **where** MANDAL\_NAME **NOT** **LIKE** '%A'

---BETWEEN OPERATION

**select**\***from** FA\_INSURANCE **where** AGE **BETWEEN** 45 **AND** 57

**select**\***from** FA\_INSURANCE **where** PREAUTH\_DATE **not** **like** '%.'

----DISTINCT OPERATION--

**select** **DISTINCT** DISTRICT\_NAME **from** FA\_INSURANCE

-----UPPER,LOWER,TRIM AND STRING OPERATIONS

**select** **UPPER**(SupplierName) **AS** SUP\_NAME,**LOWER**(SupplierAddress)**AS** SUP\_ADDRESS,LEN(SupplierContactDetails) **AS** LEN

**from** FA\_SALES\_2

**select** **rtrim**(SupplierName) **AS** RTRIMO **from** FA\_SALES\_2 E---TRIMS THE EXTRA SPACES IN RIGHT SIDE

**select** **ltrim**(SUPPLIERNAME) **from** FA\_SALES\_2 E

**select** **replace**('arunagiri','giri','@') **from** FA\_SALES\_2 E

**select** **SUBSTRING**('Arunagiri',1,5) **from** FA\_SALES\_2 E

**select** **SUBSTRING**('arunagiri@307gmail.com',charindex('@','arunagiri@307gmail.com')+1,len('arunagiri@307gmail.com')) **from** FA\_SALES\_2 E

**select** **CONCAT**('arun','a','giri') FA\_SALES\_2 E--CONCATES TWO WORDS

------CASE OPERATION

**select** **case** **when** City <='Anchorage' **then** 'Truck'

**when** City ='Saint Cloud' **then** 'Air'

**else** 'others'

**end** **as** dname **from** FA\_SALES\_2

**SUM** --- ADDITION OF INTERGER NUMBER

**AVG** ---SUM DIVIEDE NUMBER OF ELEMENTS

**MIN** ---LOWEST VALUE OF THE DATA

**MAX** ---HIGHEST VALUES OF THE DATA

**COUNT** ---NUMBER OF ELEMENTS

---SQL query with all clause .

**select** **sum**(CLAIM\_AMOUNT) **as** sum\_claim **from** FA\_INSURANCE;

**select** **avg**(CLAIM\_AMOUNT) **from** FA\_INSURANCE;

**select** **min**(CLAIM\_AMOUNT) **as** **min**,**max**(CLAIM\_AMOUNT) **as** **max** **from** FA\_INSURANCE;

**select** **count**(\*) **from** FA\_INSURANCE;

**select** **count**(3) **from** FA\_INSURANCE;

**select** **count**(1)CATEGORY\_CODE **from** FA\_INSURANCE

**group** **by** CATEGORY\_CODE

**HAVING** **COUNT**(1)>1;

----all clause function

**SELECT** **COUNT**(1),CLAIM\_AMOUNT

**FROM** FA\_INSURANCE

**WHERE** HOSP\_NAME **IS** **NOT** **NULL**

**GROUP** **BY** CLAIM\_AMOUNT

**HAVING** **COUNT**(1)>2

**ORDER** **BY** CLAIM\_AMOUNT **ASC**;

----Rank functions-------

---scenario to check which hospital is having highest claim amount using rank

**SELECT** **RANK** () **OVER** (**PARTITION** **BY** HOSP\_NAME **ORDER** **BY** CLAIM\_AMOUNT **DESC**) **AS** DRNK,HOSP\_NAME,CLAIM\_AMOUNT

**FROM** fa\_insurance

---CHECHKING DENSE RANK

**SELECT** \* **FROM** (

**SELECT** **DENSE\_RANK** () **OVER** (**PARTITION** **BY** PID **ORDER** **BY** CLAIM\_AMOUNT **DESC**) **AS** DRNK,PID,CLAIM\_AMOUNT

**FROM** fa\_insurance

**WHERE** PID = 454

)X

**WHERE** drnk = 1;

---ROW NUMBER

**SELECT** **ROW\_NUMBER** () **OVER** (**PARTITION** **BY** HOSP\_NAME **ORDER** **BY** CLAIM\_AMOUNT **DESC**) **AS** ROWNUMBER,HOSP\_NAME,CLAIM\_AMOUNT

**FROM** fa\_insurance

**DROP** **TABLE** Incurance\_Claim\_State\_Healths

**TRUNCATE** **TABLE** Incurance\_Claim\_State\_Health

**DELETE** CarSalesShipment\_FA **WHERE** SUPPLIERID = **NULL**