*DATA WAREHOUSE PROJECT*

**-Ruchika Dhungana**

Table of Contents

[**Project Introduction** 2](#_Toc59573631)

[**Aims/Objectives** 2](#_Toc59573632)

[**Data Warehouse** 2](#_Toc59573633)

[**Main Objective:** 2](#_Toc59573634)

[**OLAP:** 3](#_Toc59573635)

[**OLTP:** 3](#_Toc59573636)

[**OLAP Vs OLTP:** 3](#_Toc59573637)

[**KPI 2: Ensure customer satisfaction** 4](#_Toc59573638)

[**Task 1** 4](#_Toc59573639)

[**REPORTS:** 4](#_Toc59573640)

[**STAR SCHEMA:** 5](#_Toc59573641)

[**DATA DICTIONARY:** 5](#_Toc59573642)

[**REPORT:** 9](#_Toc59573643)

[**Task 2** 11](#_Toc59573644)

[**STAGING AREA SETUP:** 11](#_Toc59573645)

[**STAR SCHEMA SETUP:** 14](#_Toc59573646)

[**Task 3** 15](#_Toc59573647)

[**EXTRACTION:** 15](#_Toc59573648)

[**DATA PURIFICATION:** 18](#_Toc59573649)

[**TRANSFORMATION:** 25](#_Toc59573650)

[**LOAD:** 28](#_Toc59573651)

[**Task 4** 31](#_Toc59573652)

[**Data Analysis:** 31](#_Toc59573653)

[**Dashboard:** 33](#_Toc59573654)

[**Task 5** 33](#_Toc59573655)

[**Data Warehouse Approaches** 33](#_Toc59573656)

[**Assignment Portfolio** 36](#_Toc59573657)

# **Project Introduction**

The project is based off an airline company named FlyU. They hold records related to their flights such as: departure times, arrival times, customers on each flight, number of flights to reach their destination, return flights, tail number, complaints raised, and any type of flight delays.

The company stores using Oracle as well as Excel sheets. They handle these day to day data in order to plan, manage their customer complaints service and deliver a quality service.

# **Aims/Objectives**

FlyU company aims for the following results:

* Integrate a Data Warehouse to store the information relating to FlyU.
* Deliver a quality service
* Increase customer satisfaction
* Grow the company.

# **Data Warehouse**

Before we move further into the project, it is crucial to understand what data warehouse is. So, in simple terms, it is a type of data management system that stores a company’s data from one or more sources. The main purpose of data warehouse is to compare and analyse data for greater corporate performance. When it comes to business intelligence, it is considered one the vital components as it helps support business decisions by providing analytical techniques and an overall broader insight into the performance of a company. It is designed to execute query and analysis on historical data derived from heterogeneous sources.

## **Main Objective:**

These are the main objective of implementing a Data Warehouse into a company:

* **Improve quality of data**:

One of the main purpose of Data Warehouse is to guarantee data quality. Any bad or error data are analysed, purified, and transformed into a useable data hence ensuring good data quality.

* **Minimize inconsistent reports:**

Since, inconsistent reports are mainly caused by misuse of data, and the main reason for misuse of data is disagreement or misunderstanding of the meaning or the content of data. Data Warehouse ensures that there is no disagreements or misunderstandings.

* **Integrate data from multiple sources:**

**Another prime objective of Data Warehouse is to make it easy for companies to integrate data from multiple sources.**

* **Merge historical data with current data:**

As source systems do not usually keep a history of certain data, typical data warehouse objective is to store history. In data warehouse data changes in the source system are recorded, which enables historical analysis.

## **OLAP:**

**OLAP** (online analytical processing) is a core component of data warehouse which enables multidimensional data analysis for business intelligence (BI) and greater corporate performance.

OLAP cubes enable four basic types of multidimensional data analysis:

* **Drill-down:** The drill-down operation converts less-detailed data into more-detailed data
* **Roll-up:** Roll up is the opposite of the drill-down function as it aggregates detailed data.
* **Slice and Dice:** The slice operation creates a sub-cube by selecting a single dimension and the dice operation isolates a sub-cube by selecting several dimensions.
* **Pivot:**The pivot function rotates the current cube view to enable dynamic multidimensional views of data.

## **OLTP:**

**OLTP**(*online transaction processing*) enables transaction-oriented applications in a 3-tier architecture. OLTP administers day to day transaction of an organization.

## **OLAP Vs OLTP:**

The differences between OLAP and OLTP:

|  |  |
| --- | --- |
| OLAP | OLTP |
| The primary objective is data analysis. | The primary objective is data processing. |
| OLAP can be used for all type of business analysis needs which includes planning, budgeting, forecasting, and analysis | OLTP is useful to administer day to day transactions of an organization. |
| OLAP uses data warehouse technique where it can integrate different data sources for building a secure database. | OLTP uses traditional DBMS |

# **KPI 2: Ensure customer satisfaction**

As FlyU company is looking to integrate a Data Warehouse to focus on their **Key Performance Indicators (KPI) which includes** deliver a quality service, increase customer satisfaction and grow the company. We will be focusing on the second KPI i.e. ensure customer satisfaction.

This KPI will focus on all the data that are necessary to make sure that the company will be able to reach a decision on how to increase their customer satisfaction, based on the analysis report that we will create using OLAP system.

## **Task 1**

### **REPORTS:**

These are the reports that will be generated once the project is completed.

* Identify the airlines with most complaints.
* The no. of complaints per flight.
* Total amount of compensation given each month in the year 2017.
* The no. of complain per flight per month in the year 2017.
* The total number of compensation given in 2017, according to the type of complaints.

### **STAR SCHEMA:**

Diagram

Description automatically generated

### **DATA DICTIONARY:**

Data dictionary provides a clear picture about the contents, format, and structure of a database and the relationship between its elements as well as any bad data that need to be transformed.

* DIM\_TIME

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Star Schema**  **Table** | **Attribute Name** | **Data Type** | **Key** | **DQ Source** | **Data**  **Mapping** | **Data quality Issues** | **Transformation** |
| DIM\_TIME | timeID | Integer | Yes | Automatically generated as a primary key | n/a | n/a | Create a sequence timeid\_seq to generate primary keys |
| year | Number | No | FlyU\_flights | n/a | n/a | n/a |
| month | Number | No | FlyU\_flights | n/a | n/a | n/a |
| day | Number | No | FlyU\_flights | n/a | n/a | n/a |
| Definition: | The dim\_time table holds the intervals of time for which the data will be held. It is held at year, month, day and quarterly level meaning. | | | | | | |
| Notes: |  | | | | | | |

* DIM\_FLIGHTS

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Star Schema**  **Table** | **Attribute Name** | **Data Type** | **Key** | **DQ Source** | **Data**  **Mapping** | **Data quality Issues** | **Transformation** |
| DIM\_FLIGHTS | flightKey | INTEGER | Yes | Automatically generated as surrogate key | n/a | n/a | Create a sequence flight\_seq to generate primary keyS |
| flightNo | INTEGER | No | FlyU\_flights | n/a | n/a | n/a |
| TailNo | INTEGER | No | FlyU\_flights | n/a | n/a | n/a |
| dbSource | VARCHAR | No | Should be generated | n/a | n/a | Create a sequence SOURCE\_seq to generate Quarterly dates |
| Definition: | The dim\_FLIGHTS table holds the data related to all the flights. | | | | | | |
| Notes: |  | | | | | | |

* DIM\_COMPENSATION

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Star Schema**  **Table** | **Attribute Name** | **Data Type** | **Key** | **DQ Source** | **Data**  **Mapping** | **Data quality Issues** | **Transformation** |
| DIM\_  COMPENSATION | compensationKey | INTEGER | Yes | Automatically generated as surrogate key | n/a | n/a | Create a sequence comp\_seq to generate primary keys |
| compensation\_amnt | INTEGER | No | FlyU\_flights | n/a | n/a | n/a |
| compensation\_type | VARCHAR | No | FlyU\_flights | Null value | Some complaints are missing the  compensation type | Will need to mention the compensation type |
| Inconsistent Data | Inconsistency in compensation type. Eg: ‘rebooked’, ‘rebook’. | Will need to convert the compensation type from ‘rebook’ to rebooked. |
| dbSource | VARCHAR | No | Should be generated | n/a | n/a | Create a sequence SOURCE\_seq to generate quarterly dates |
| Definition: | The dim\_compensation table holds the data related to the compensation given to the customers. | | | | | | |
| Notes: |  | | | | | | |

* DIM\_COMPLAINTS

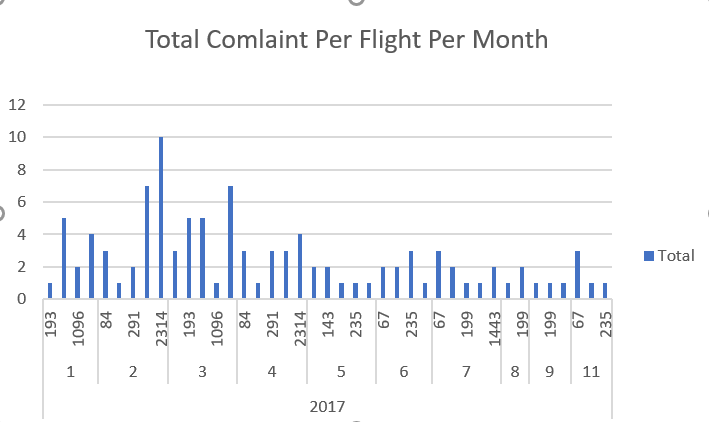
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Star Schema**  **Table** | **Attribute Name** | **Data Type** | **Key** | **DQ Source** | **Data**  **Mapping** | **Data quality Issues** | **Transformation** |
| DIM\_  COMPLAINTS | Complaint  Key | INTEGER | Yes | Automatically generated as surrogate key | n/a | n/a | Create a sequence complain\_seq to generate primary keys |
| Complaint  Id | INTEGER | No | FlyU\_flights | n/a | n/a | n/a |
| Complaint  Type | VARCHAR | No | FlyU\_flights | Null value | Some complaints are missing the  compensation type | Will need to add the missing compensation type |
| Inconsistent value | Complaint types have irregural values. Eg: A,B,C for cancellation | Will need to transform all the irregular values to ‘C’ for cancellation and ‘L’ for late. |
| Description | VARCHAR | No | FlyU\_flights | Null Value | Some complaints are missing the  description | Will need to add the missing description |
| Complaint  status | VARCHAR | No | FlyU\_flights | n/a | n/a | n/a |
| dbSource | VARCHAR | No | Should be generated | n/a | n/a | Create a sequence SOURCE\_seq to generate quarterly dates |
| Definition: | The dim\_complaint table holds all the data related to the customer complaints. | | | | | | |
| Notes: |  | | | | | | |

* FACT\_COMPLAINTS

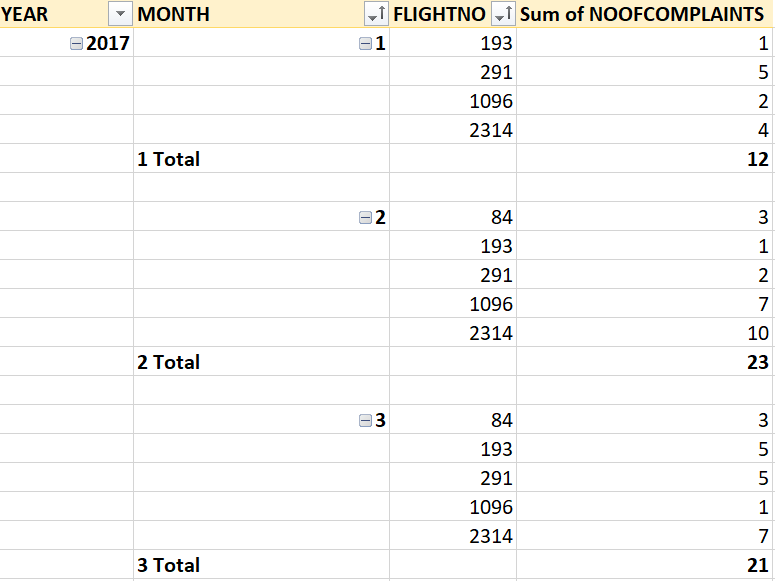
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Star Schema**  **Table** | **Attribute Name** | **Data Type** | **Key** | **DQ Source** | **Data**  **Mapping** | **Data quality Issues** | **Transformation** |
| FACT\_  COMPLAINTS | FactId | INTEGER | Yes | Automatically generated as primary key | n/a | n/a | Create a sequence fact\_seq to generate primary keys |
| Complaint  Key | INTEGER | No | FlyU\_flights | n/a | n/a | n/a |
| Compensation  Key | INTEGER | No | FlyU\_flights | n/a | n/a | n/a |
| TimeId | INTEGER | No | FlyU\_flights | n/a | n/a | n/a |
| flightKey | INTEGER | No | FlyU\_flights | n/a | n/a | n/a |
| noOfComplaint | INTEGER | No | Should be generated | n/a | n/a | Create a sequence SOURCE\_seq to generate quarterly dates |
| total  Compensation | INTEGER | No | Should be generated | n/a | n/a | Create a sequence SOURCE\_seq to generate quarterly dates |
| Definition: | The dim\_complaint table holdsall the data related to the customer complaints. | | | | | | |
| Notes: |  | | | | | | |

### **REPORT:**

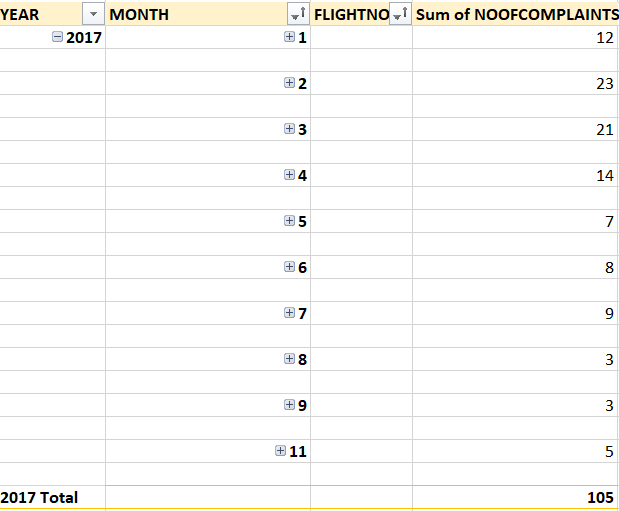
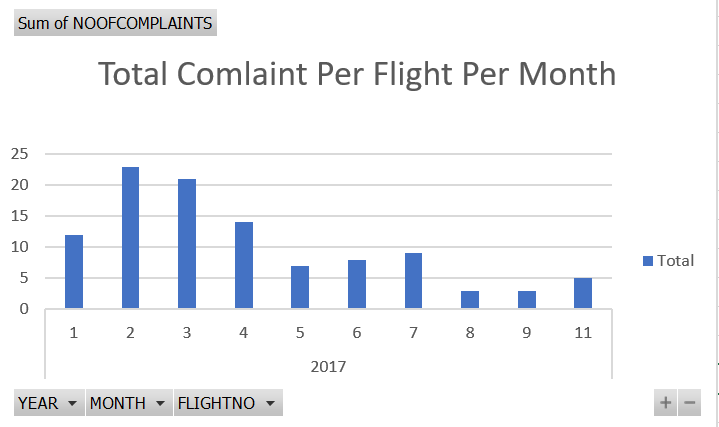
* EXCEL REPORT



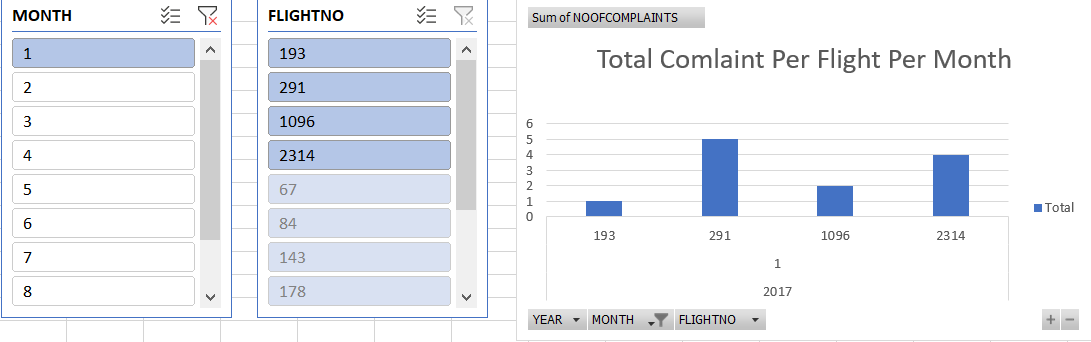
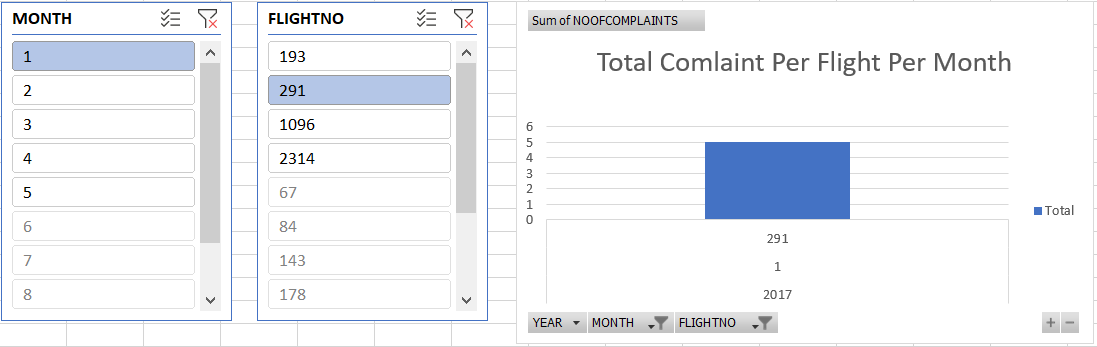
* Drill Down Report



* Rollup Report

* SLICING/DICING

## **Task 2**

### **STAGING AREA SETUP:**

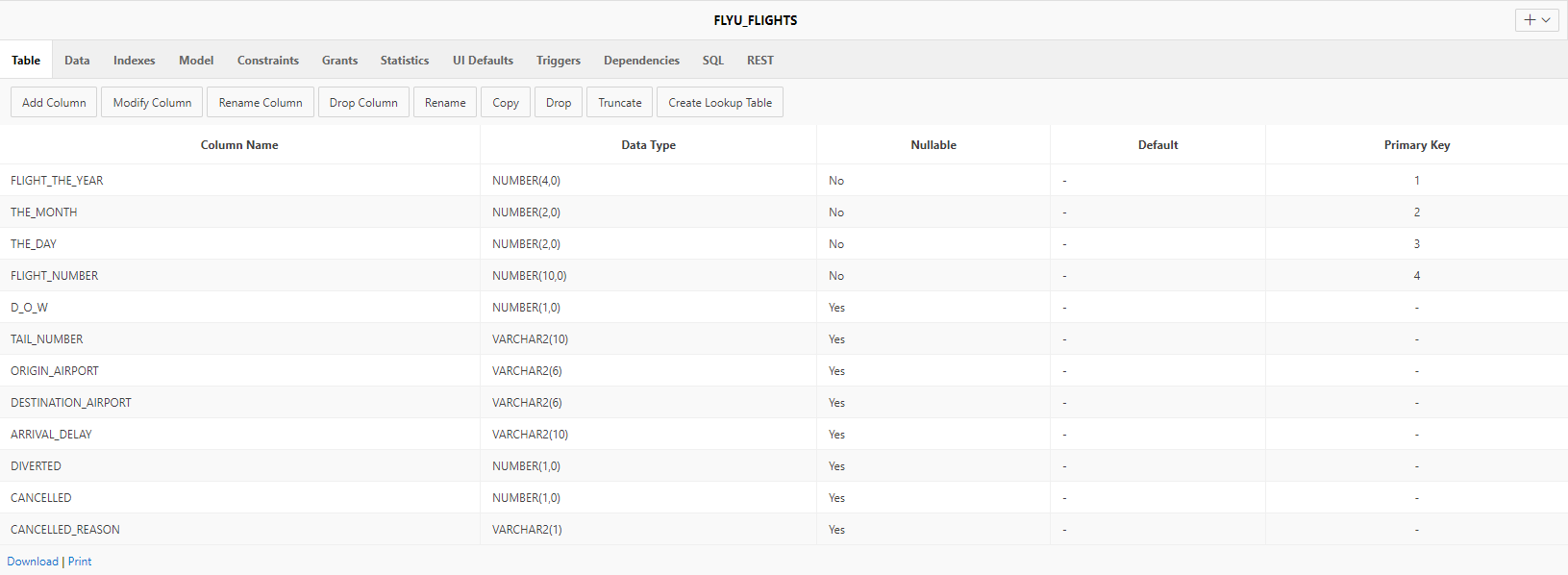
In this step, the multiple sources of data are extracted and run into the database. In this case, as we are dealing with complaints and customer service, we have only extracted flyU\_flights data source. Hence we ended up with three tables: FlyU\_flights, Complaints and Customers.

* Data Source

Graphical user interface, text, application, email

Description automatically generated

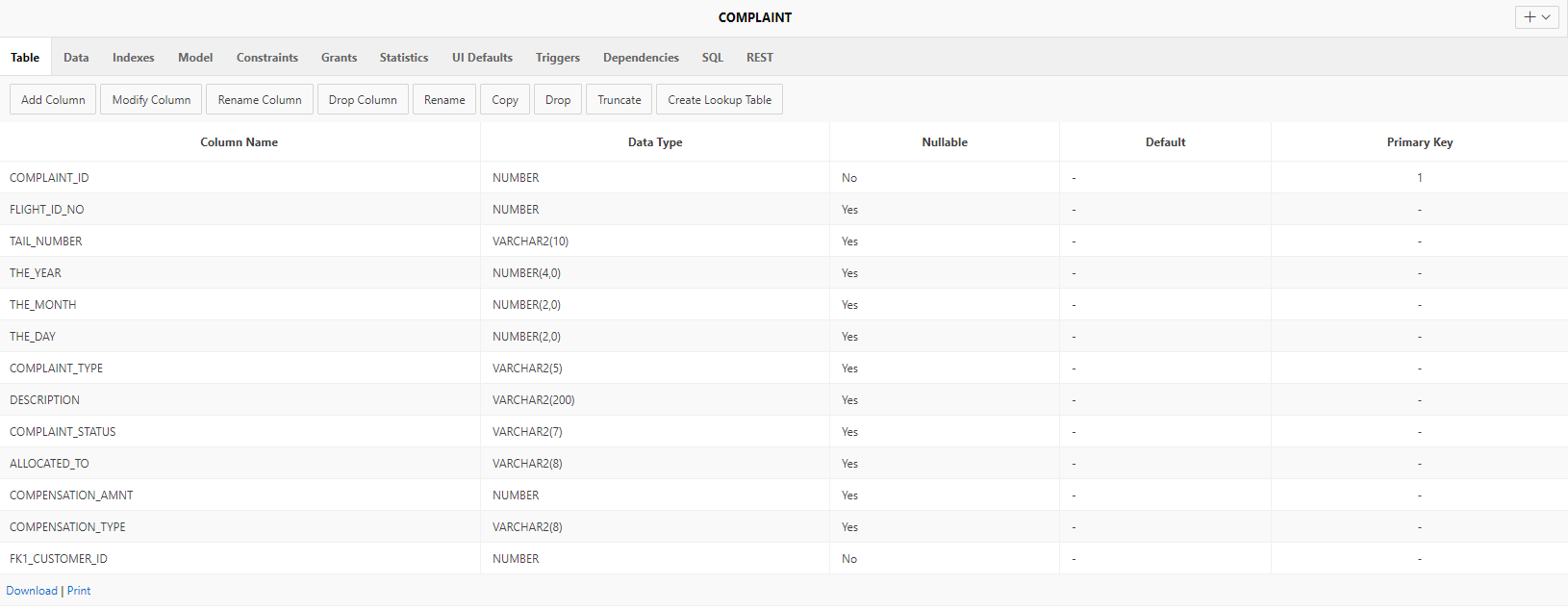
* FlyU\_flights

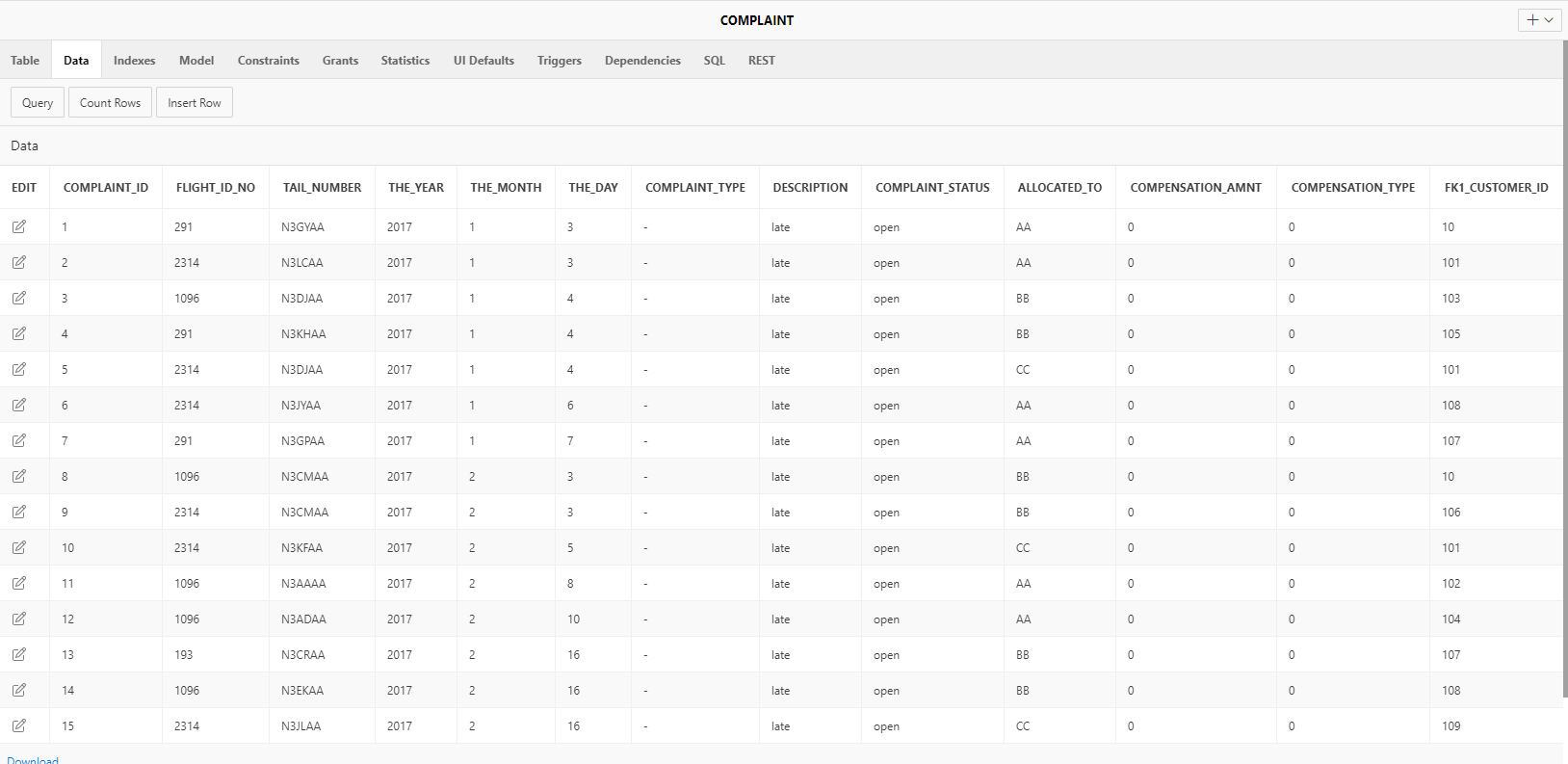


Table

Description automatically generated

* Complaint

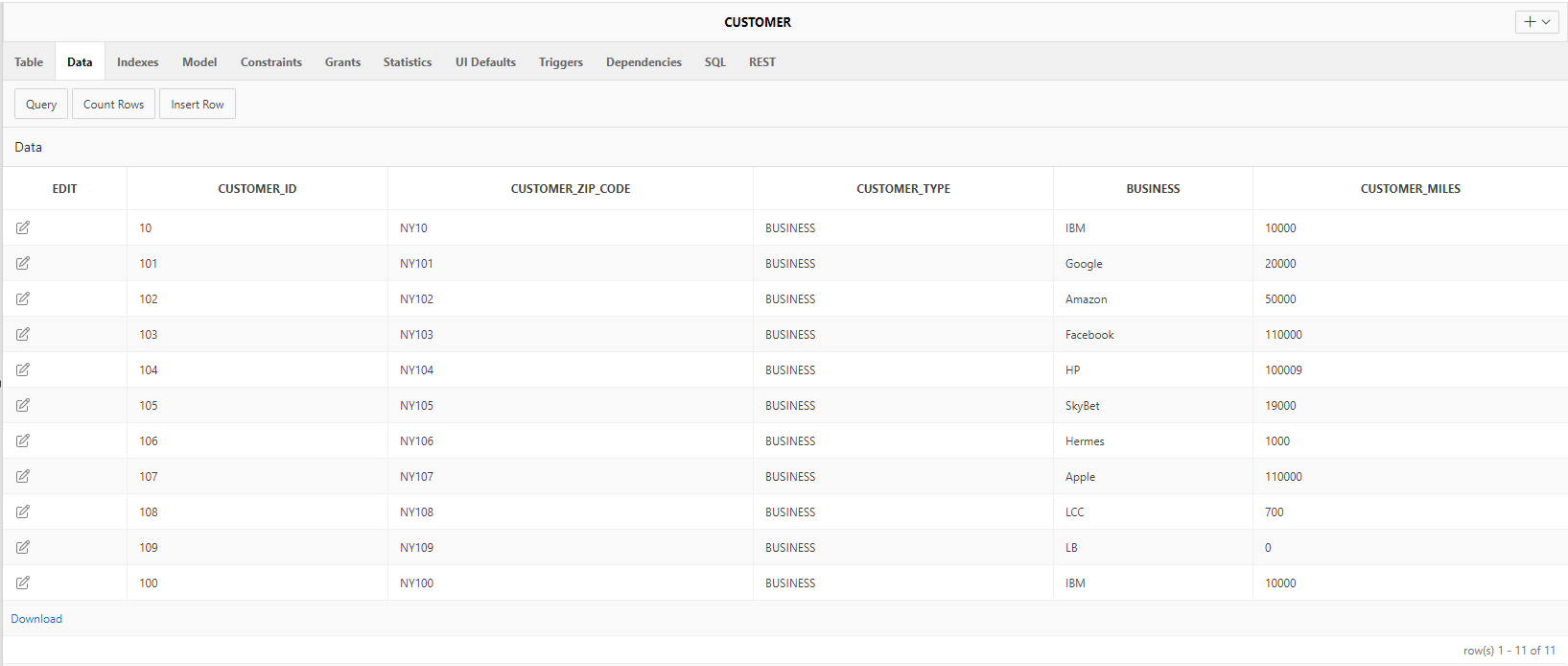




* Customer

Graphical user interface, text, application, chat or text message

Description automatically generated



### **STAR SCHEMA SETUP:**

We have created a star schema based on the reports created.

Diagram

Description automatically generated

The script of the star schema is forward engineered using Qsee tool and uploaded into the database.

Graphical user interface, text, application, email

Description automatically generated

## **Task 3**

This part of the project deals with building the data warehouse using the **ETL process**. It is a type of data integration completed in three steps: extract, transform, load.

### **EXTRACTION:**

In this part of the ETL process, we have extracted data from the data sources and transferred into the staging area.

Graphical user interface, text, application

Description automatically generated

There are three staging tables:

* **Stage Complaints**

Graphical user interface, application

Description automatically generated

Graphical user interface, table

Description automatically generated

* **Stage Compensation**

Graphical user interface, application, Teams

Description automatically generated

Table

Description automatically generated

* **Stage Flights**

Graphical user interface, application

Description automatically generated

Graphical user interface, application, table, Teams

Description automatically generated

### **DATA PURIFICATION:**

In this part of the ETL process, we identify all the bad and good data’s determined in the data dictionary, create good and bad tables and populate it, reclean bad data and transfer the bad data into the good tables.

COMPLAINT

* Table Creation

Graphical user interface, text, application, Word, email

Description automatically generated

* Bad/Good Data IdentificationGraphical user interface, text, application, email

  Description automatically generated

Graphical user interface, application, Teams

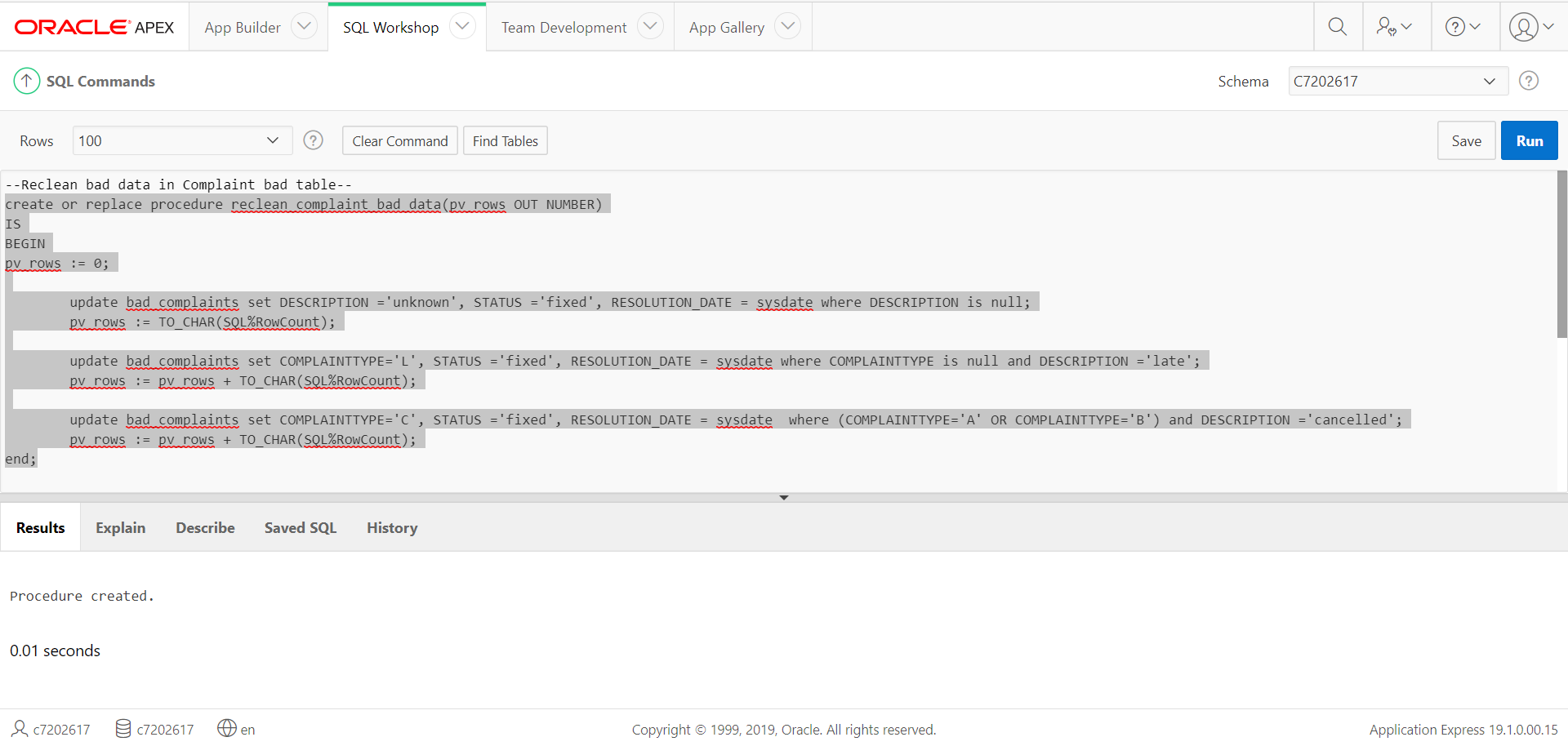
Description automatically generated

* BAD TABLE

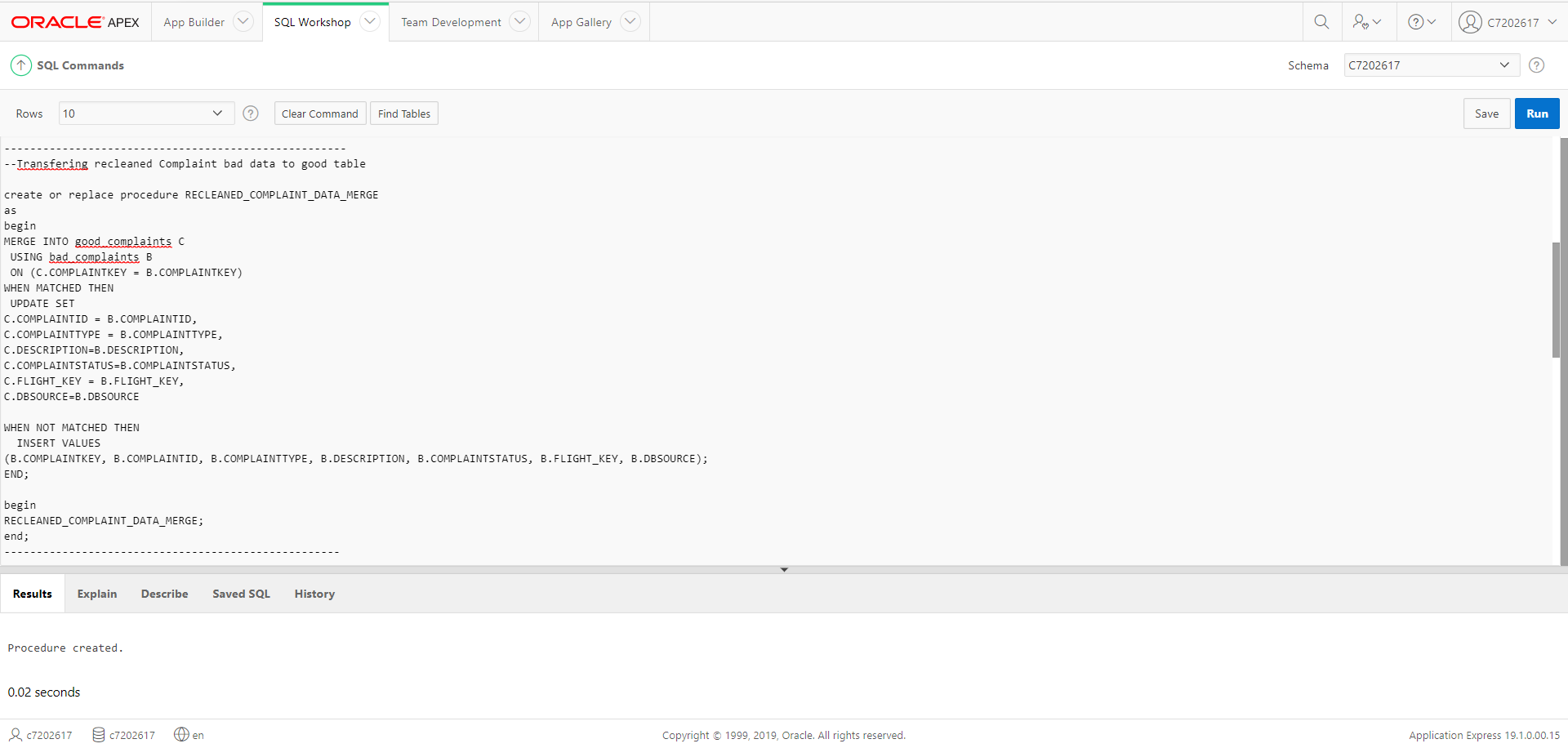
Table

Description automatically generated

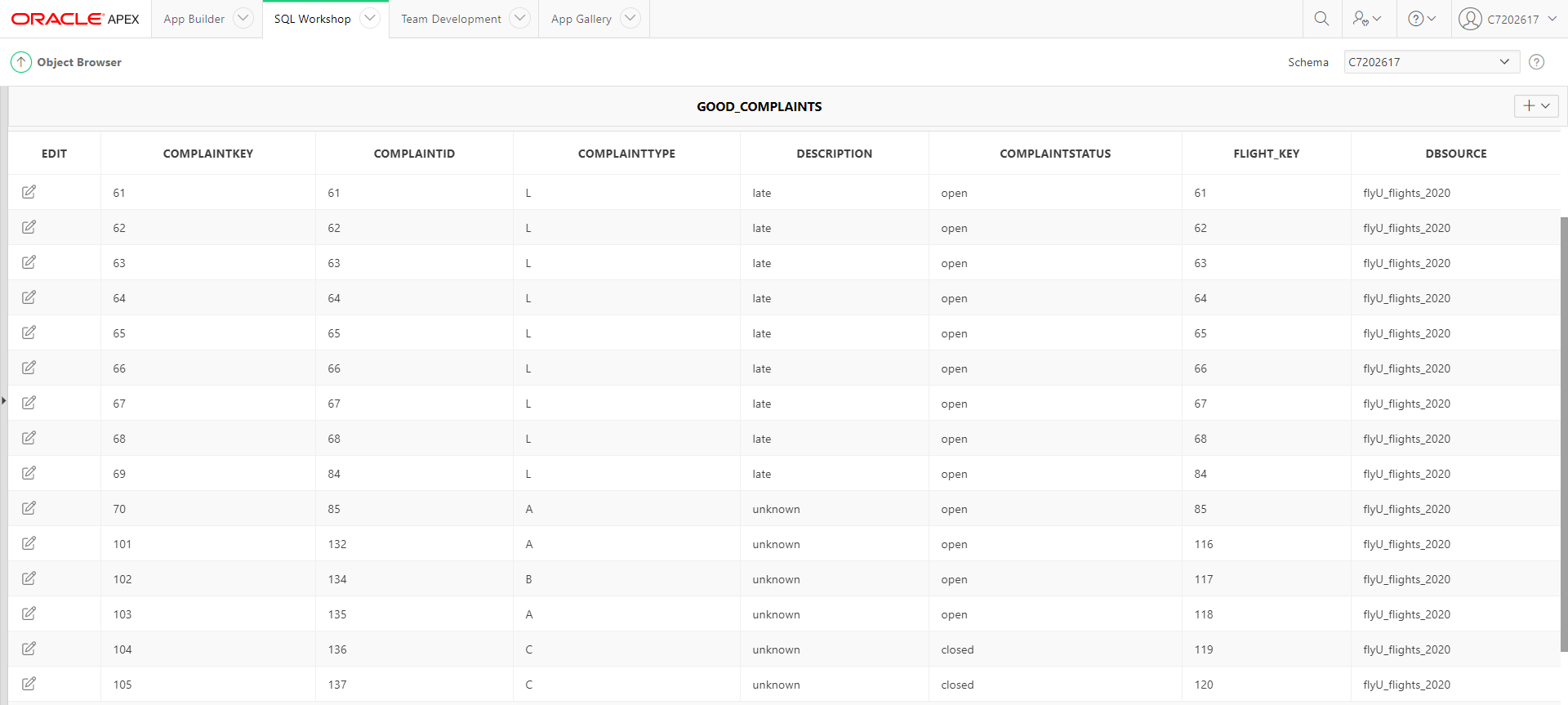
* Reclean Bad Data



* Transfer Recleaned Data

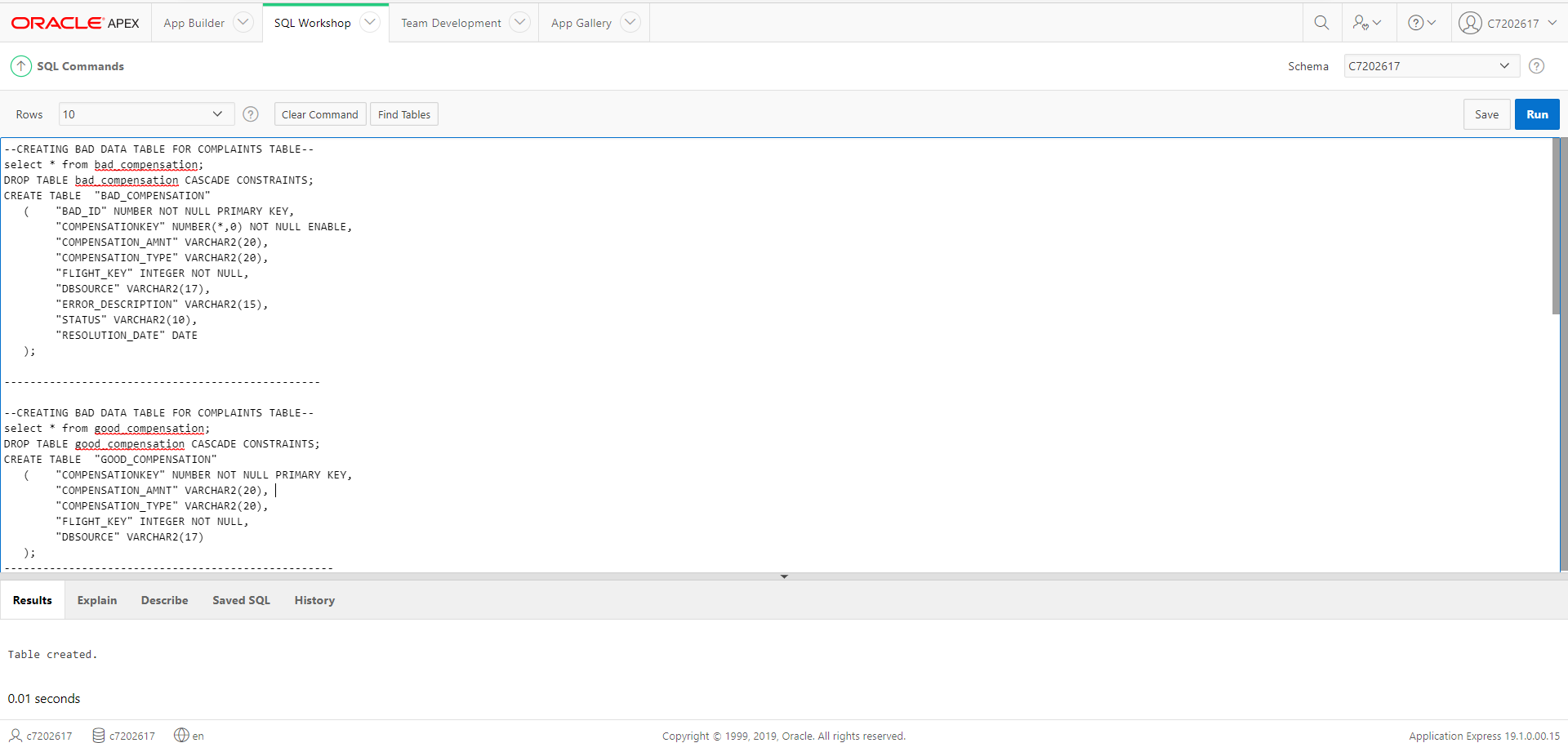


* Good Table

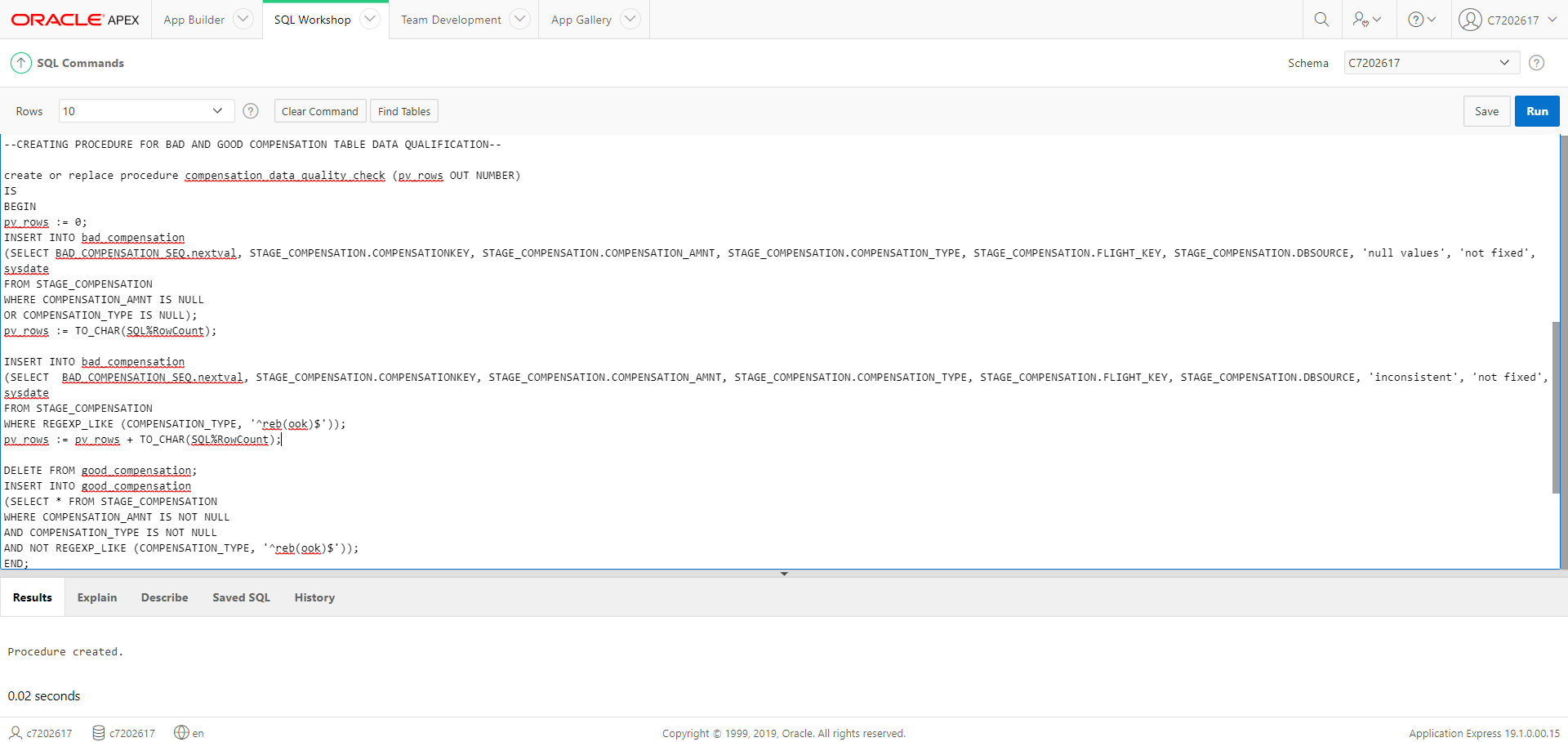


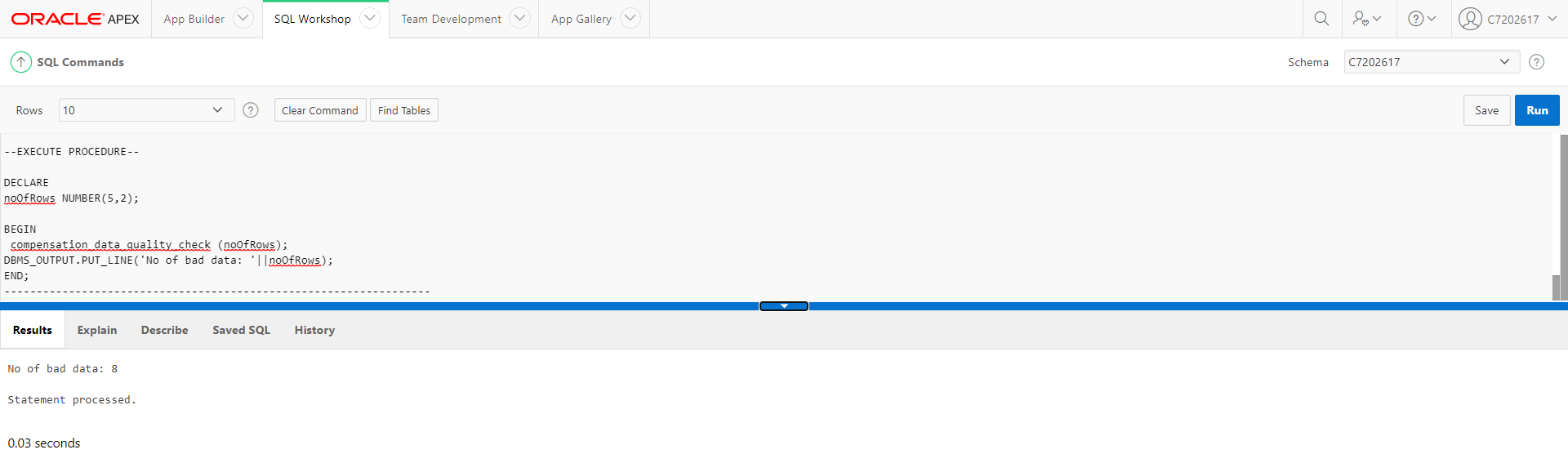
COMPENSATION

* Table Creation

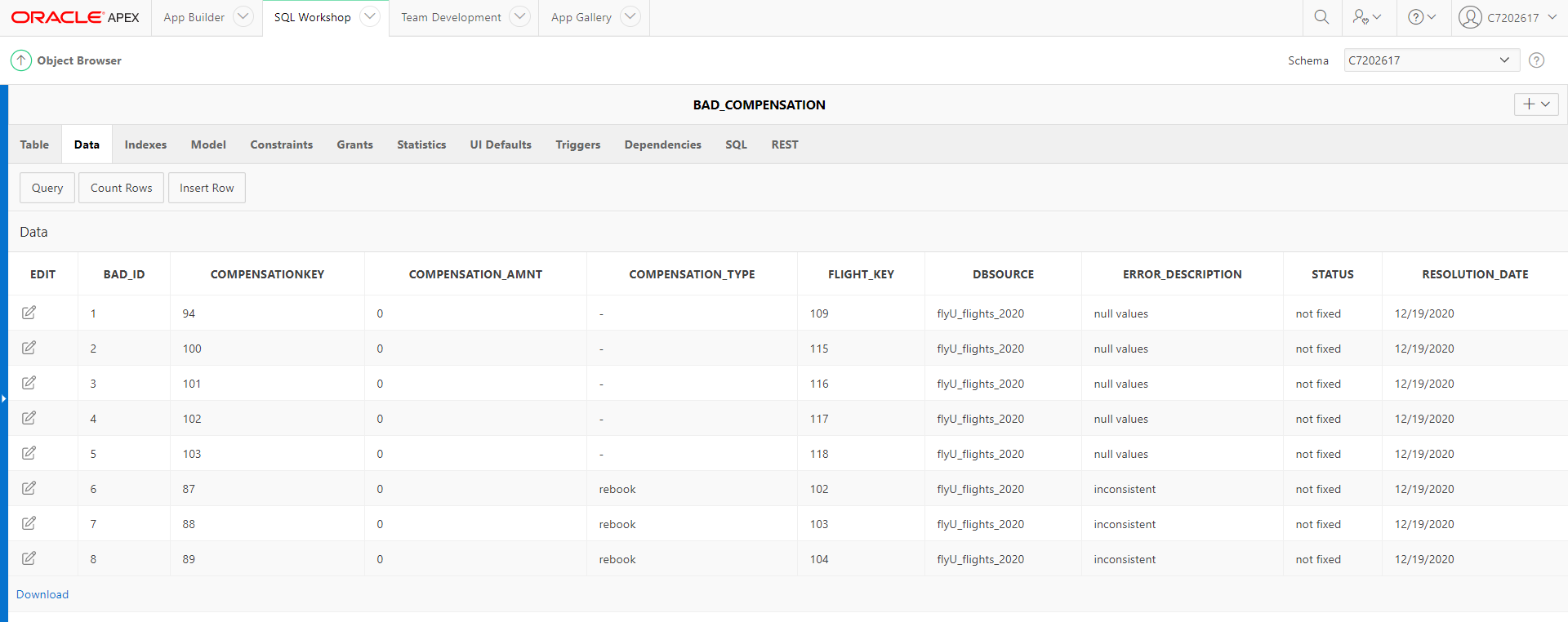


* Bad/Good Data Identification

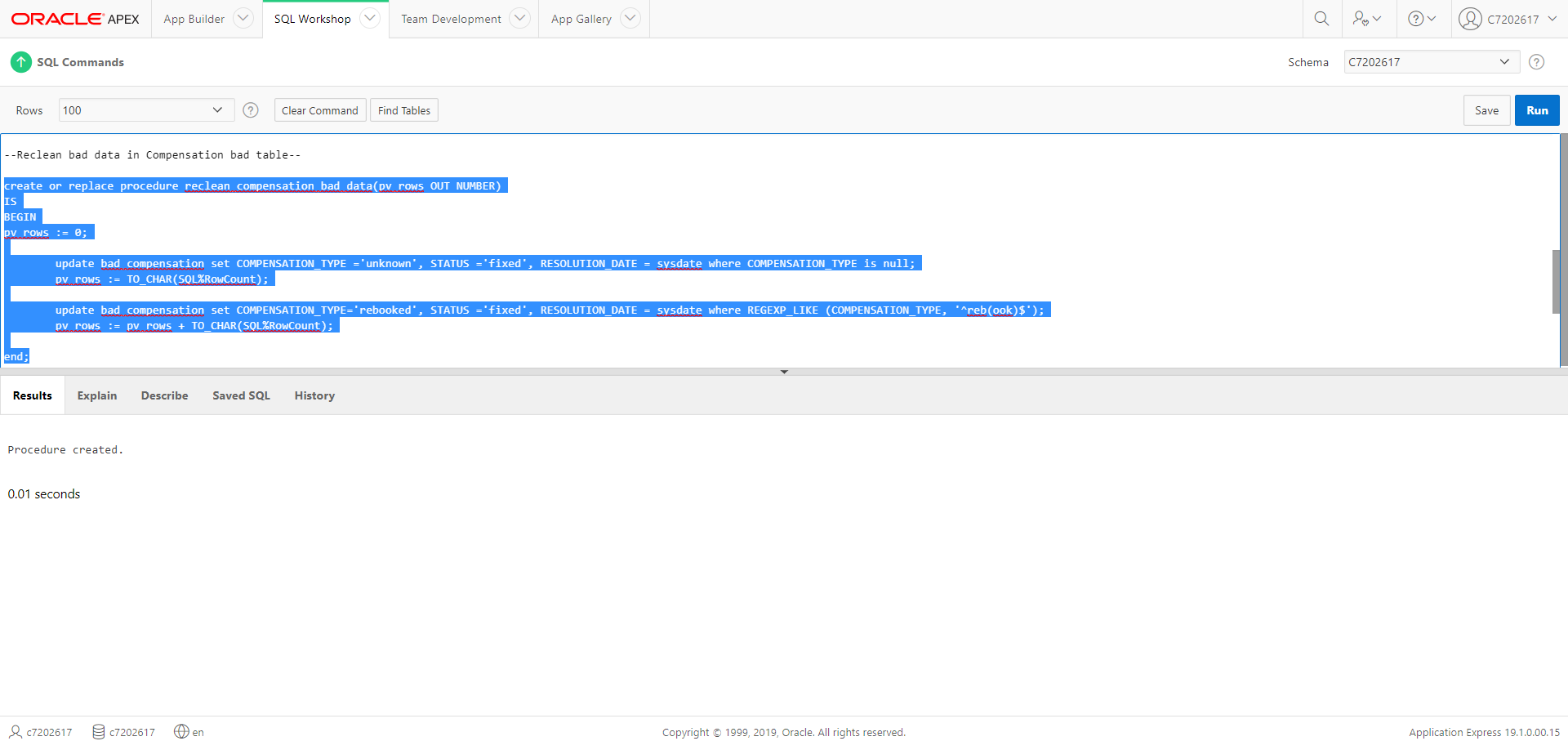


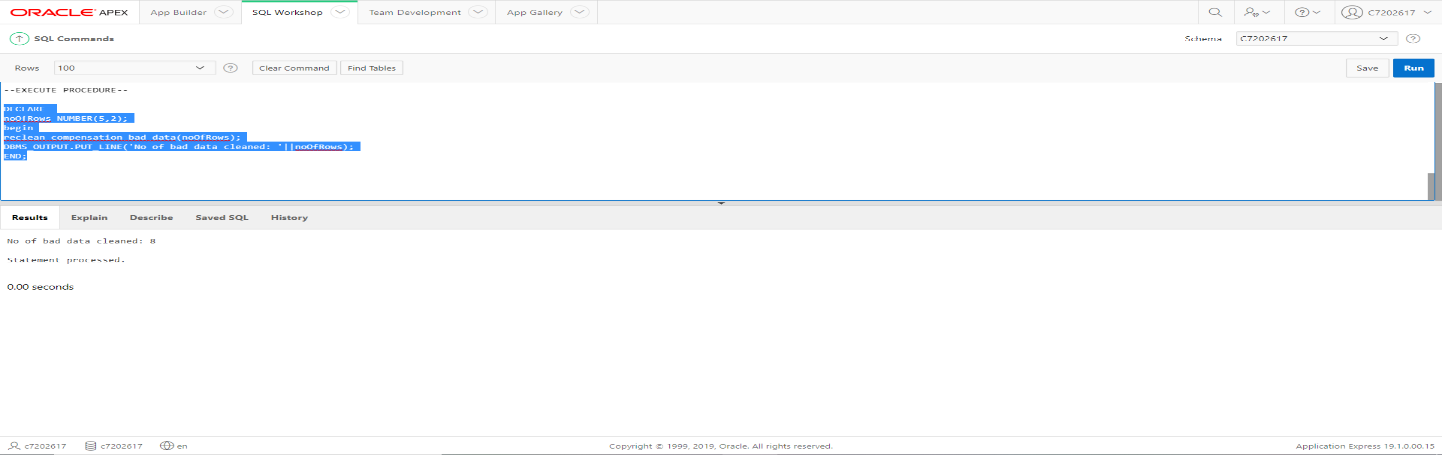


* BAD TABLE

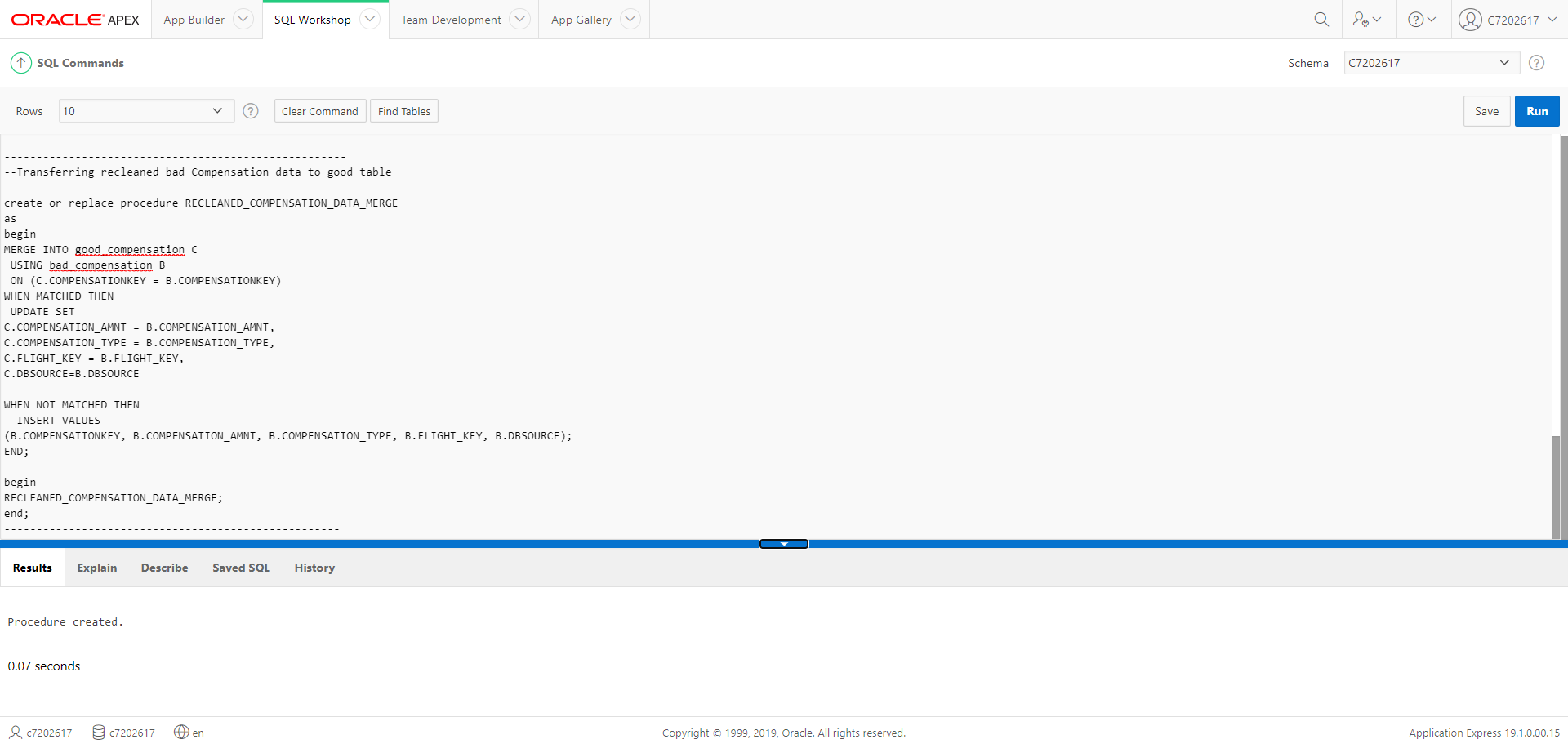


* Reclean Bad Data

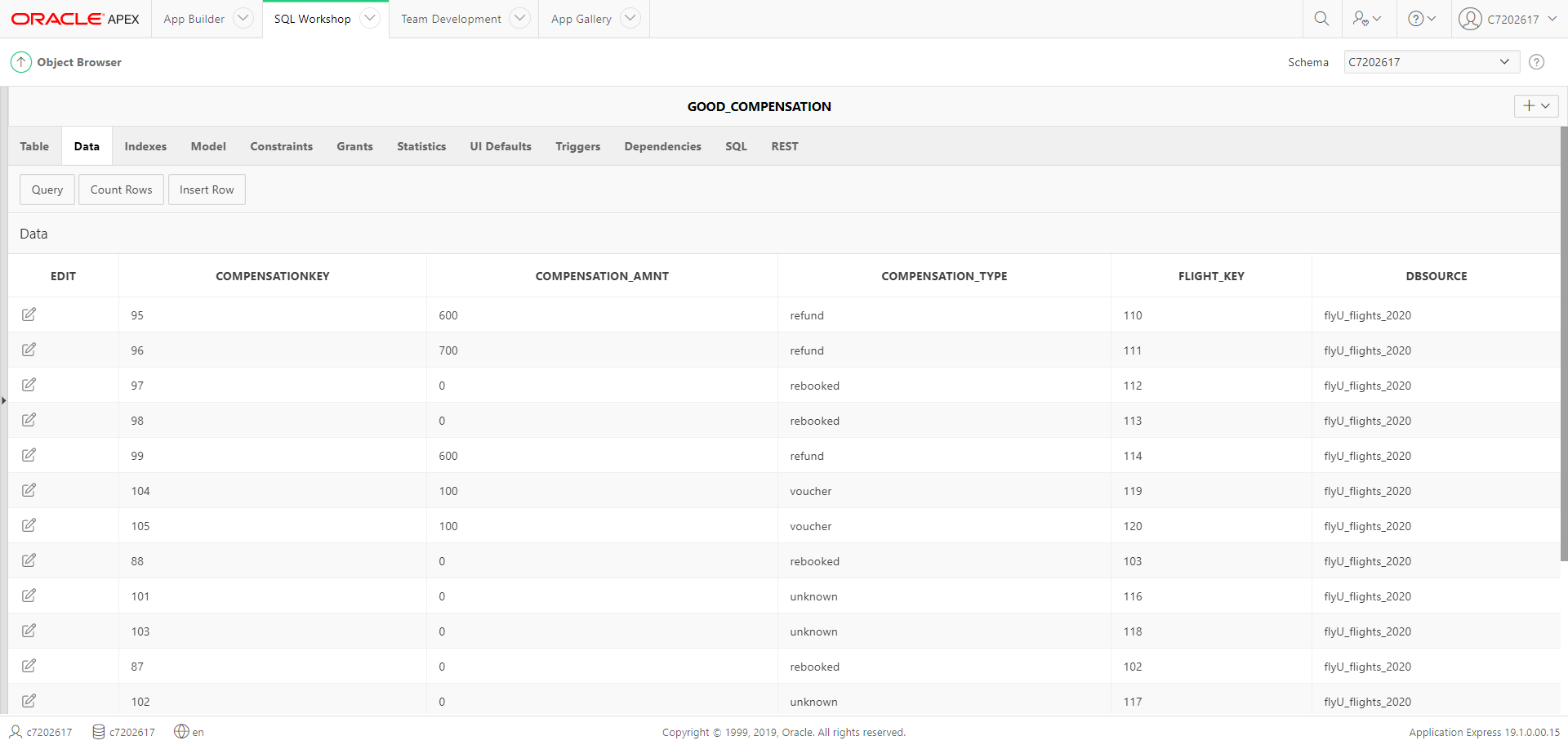




* Transfer Recleaned Data

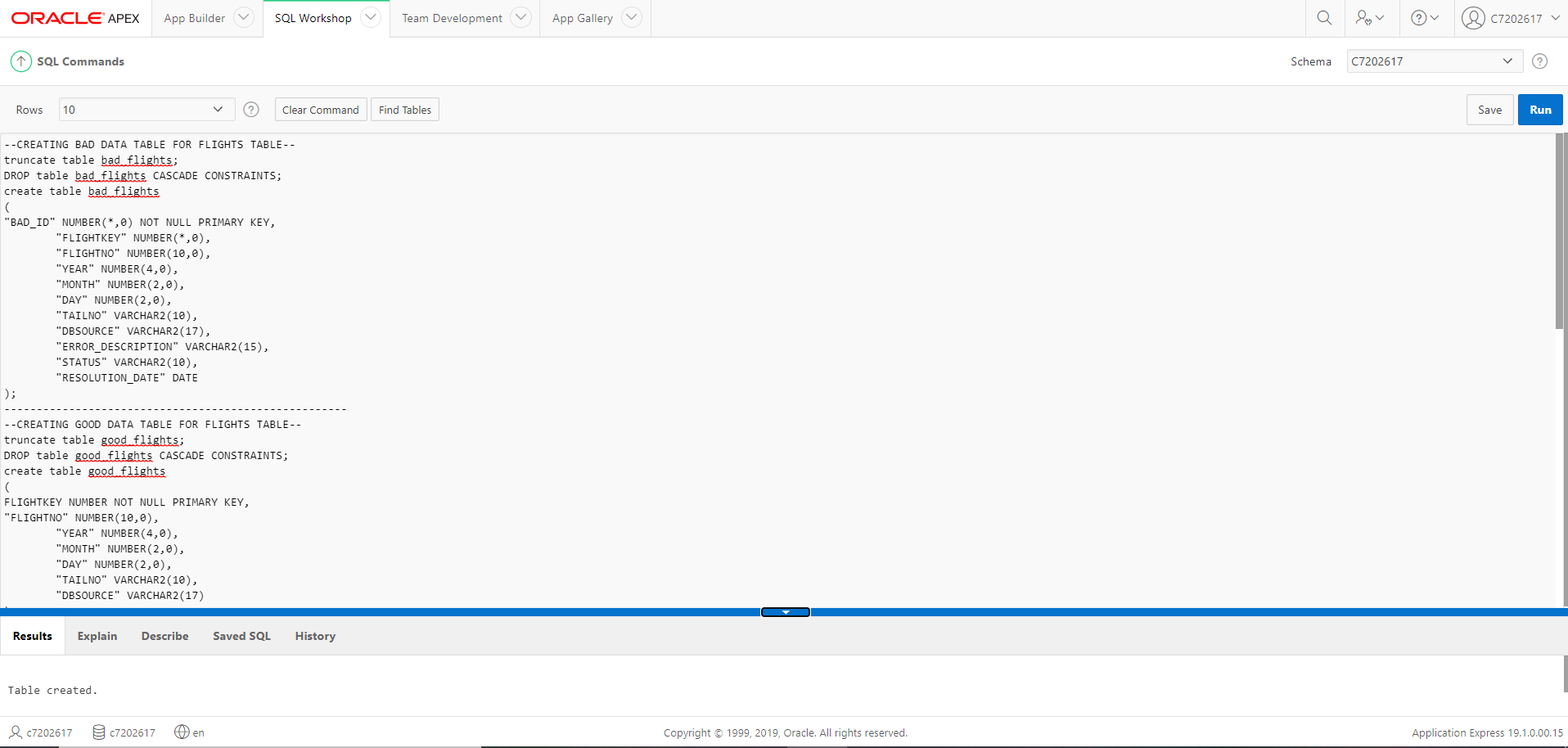


* Good Table

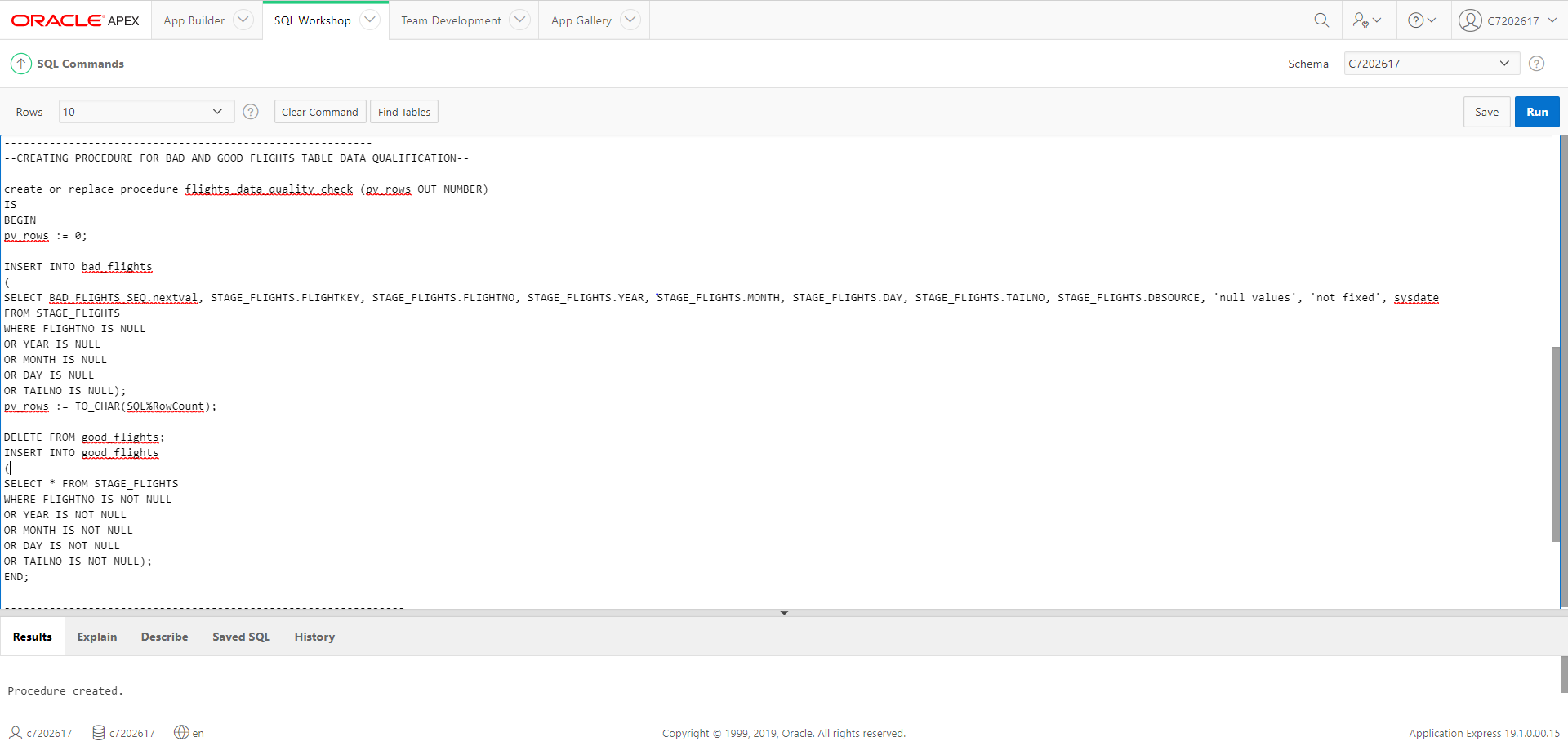


FLIGHTS

* Table Creation



* Bad/Good Data Identification



* BAD TABLE

Graphical user interface, text, application, email

Description automatically generated

* Reclean Bad Data

There is no bad data to be cleaned for flights table.

### **TRANSFORMATION:**

In this part of the ETL process, we transform all the cleaned data from the clean tables into the their respective transformation table.

* **Transform Complaints**

Graphical user interface, application, Word

Description automatically generated

Graphical user interface, application, Word

Description automatically generated

Table

Description automatically generated

* **Transform Compensation**

Graphical user interface, application, Word

Description automatically generated



Table

Description automatically generated

* **Transform Flights**

Graphical user interface, application, Word

Description automatically generated

Graphical user interface, text, application, Word

Description automatically generated

Table

Description automatically generated

### **LOAD:**

In this part of the ETL process, we load all the data from the transformation table into the star schema design we upload above in task 1.

Graphical user interface, text, application, email

Description automatically generated

These are the dimension table with quality data that can be analyzed:

* **DIM\_TIME**

Table

Description automatically generated

* **DIM\_FLIGHTS**

Table

Description automatically generated

* **DIM\_COMPLAINTS**

Table

Description automatically generated

* **DIM\_COMPENSATION**

Graphical user interface, table

Description automatically generated

* **FACT\_COMPLAINTS**

Graphical user interface, text, application, email

Description automatically generated

Table

Description automatically generated

## **Task 4**

### **Data Analysis:**

* Report 1

Chart

Description automatically generated

* Report 2

Chart, line chart

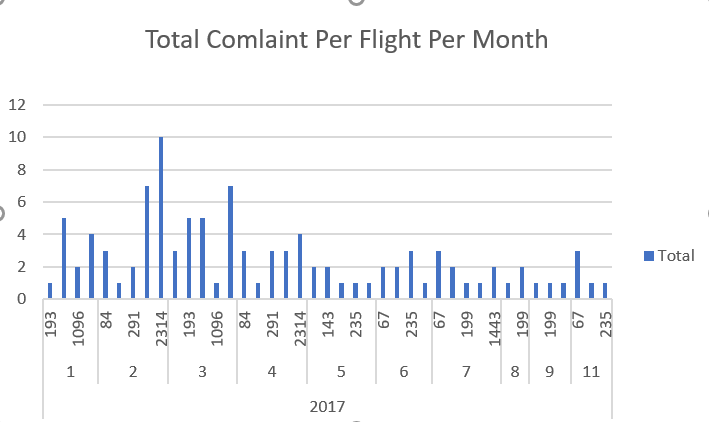
Description automatically generated

* Report 3

Chart, pie chart

Description automatically generated

* Report 4



* Report 5

Chart, line chart

Description automatically generated

### **Dashboard:**

This dashboard facilitates the company to make multi-dimensional analysis based on flights, complaints and time.

Graphical user interface, chart, application

Description automatically generated

## **Task 5**

### **Data Warehouse Approaches**

In Data Warehouse, there are two major approaches when it comes to designing and they are:

* Inmon Method
* Kimball Method

When it comes to effective corporate performance, it is crucial to determine the appropriate approach according to the requirement of the project. This helps cut down project cost as well as save a lot of time. Both methods have their own advantages and differentiating factors, so determining which method to use determines the future of the company.

**Bill Inmon’s Method**

The Bill Inmon’s architecture, also known as the top-down design, organizes data using ER modelling. In this architecture, a normalised data model is designed before the dimensional data marts where all the required data are created from the data warehouse. The method enacts data warehouse as a centralised repository where it captures the “atomic” data at the very lowest level of detail hence, earning the name, atomic data warehouse. Therefore, it provides a logical framework for delivering business intelligence as it is at the centre of the corporate information factory (CIF). Simply, it is starting with building a big, centralized enterprise data warehouse where all available data from transaction systems are consolidated into a subject-oriented, integrated, time-variant and non-volatile collection of data that supports decision making. Then data marts are built for analytic needs of departments.

The Inmon design approach uses the normalized form for building entity structure, avoiding data redundancy as much as possible. This results in clear identification of business requirements and improving any data irregularities.

**Advantages of Bill Immon’s Method:**  
The Inmon architecture offers the following advantages :

* The data warehouse acts as a centralized unit for the entire company, where data from multiple sources can be integrated.
* This approach data warehouse process is less likely to result failure as it avoids data redundancy as much as possible resulting in relatively less data irregularities.
* As the top-down model represents data at a very lowest level of detail, making decision making and analytical process simpler.
* This approach is greatly flexible, as it is easier to update the data warehouse in case there is any change in the data sources , time, business requirements, etc.
* It can handle diverse enterprise-wide reporting requirements.

**Disadvantages of Bill Immon’s Method:**

The Inmon architecture offers the following disadvantages :

* It can be susceptible to more complexity because over time, multiple tables are added to the data warehouse.
* It can be expensive in terms of hiring resources skilled in data science.
* The initial setup and delivery can take a lot of time.
* Additional ETL operation is required since data marts are created after the creation of the data warehouse.

**Ralph Kimball Method**

The Ralph Kimball architecture, also known as bottom-up design of Data Warehouse(DW), forms data marts first based on the business requirements. In this architecture, the key business questions and the key business processes are identified before the primary data sources are evaluated. Once, the data sources are analysed and documented, the Extract, Transform and Load (ETL) software is utilized to fetch data from multiple sources and load into a staging area. After that, the data purification process occurs where data are segregated into clean and error table. The data in error table are then recleaned and transformed into the clean data. From here, data is loaded into a dimensional which is not normalized. The dimensional modelling is done using the star schema. In the star schema, there is typically a fact table surrounded by many dimensions. The fact table has all the measures that are relevant to the subject area, and it also has the foreign keys from the different dimensions that surround the fact. The dimensions are denormalized completely so that the user can drill up and drill down without joining to another table.

The Kimball design approach uses the denormalized form for building entity structure. It is also based on conformed facts i.e. data marts which are separately implemented are grouped together with a robust architecture.

**Advantages of Bill Immon’s Method:**  
The Inmon architecture offers the following advantages :

* The initial setup and execution is faster as there is no normalization involved.
* It simplifies querying and analysis as the data operators can be easily interpreted because of its denormalized structure.
* It takes less space in the database which makes system management simpler.
* A smaller team of designers and planners is sufficient for data warehouse management.
* It provides multi-dimensional structure and helps generate reliable insights.

**Disadvantages of Bill Immon’s Method:**

The Inmon architecture offers the following disadvantages :

* In Kimball design, data isn’t entirely integrated before reporting.
* As redundant data is added to database tables, data irregularities are most likely to occur.
* In the Kimball DW approach, the data warehouse model may be difficult to alter with any change in business needs.
* The model is business process-oriented so it won’t focus on the other areas of the enterprise.

### **Assignment Portfolio**

**Data Warehouse design for LBU business:**

* Design the star schema for the DW to be implemented.

Diagram

Description automatically generated

* Define, fact table, dimension(s), attributes, keys and measures.

**Fact:** Order Table (OrderID, customerKey, agentKey, timeID, branchKey, totalOrders)

**Dimensions:** Customer (customerKey, customerID, custName, dbSource)

Agent (agentKey, agentID, agentName, dbSource)

Branch (branchKey, branchID, branchName, dbSource)

Time (timeID, Year, Quarter, dbSource)

**Measures:** totalOrders

* Reports:

1. Number of orders made in UK first quarter of the year, in comparison with last year.

**SELECT** t.Quarter, SUM (totalOrders)

**FROM** Order o, Time t

**WHERE** o.timeID = t.timeID

**AND** t.Quarter = “Q1”

**AND TO\_CHAR**(SYSDATE, ‘YYYY’) = t.Year

**GROUP BY** t.Quarter

**UNION**

**SELECT** t.Quarter, SUM (totalOrders)

**FROM** Order o, Time t

**WHERE** o.timeID = t.timeID

**AND TO\_CHAR**(SYSDATE, ‘YYYY’) -1 = t.Year

**GROUP BY** t.Quarter’

1. Who is our best customer, in first quarter of this year?

**SELECT** SUM (totalOrders), customerKey

**FROM**

**(**

**SELECT** SUM (totalOrders), customerKey

**RANK OVER** (**ORDER BY** SUM (totalOrders) **DESC**) **AS** Rank

**FROM** Order o, Time t,

**WHERE** o.timeID = t.timeID

**AND** t.Quarter = “Q1”

**AND TO\_CHAR**(SYSDATE, ‘YYYY’) = t.Year

**)**

**WHERE** Rank <=1;

**GROUP BY** customerKey;

1. Total number of orders made in first quarter of the year, in comparison with last year for each branch?

**SELECT** t.Quarter, SUM (totalOrders), branchKey

**FROM** Order o, Time t

**WHERE** o.timeID = t.timeID

**AND** t.Quarter = “Q1”

**AND TO\_CHAR**(SYSDATE, ‘YYYY’) = t.Year

**GROUP BY** t.Quarter, branchKey

**UNION**

**SELECT** t.Quarter, SUM (totalOrders), branchKey

**FROM** Order o, Time t

**WHERE** o.timeID = t.timeID

**AND TO\_CHAR**(SYSDATE, ‘YYYY’) -1 = t.Year

**GROUP BY** t.Quarter, branchKey

1. Who is our best Agent, in the first quarter of this year?

**SELECT** SUM (totalOrders), agentKey

**FROM**

**(**

**SELECT** SUM (totalOrders), agentKey

**RANK OVER** (**ORDER BY** SUM (totalOrders) **DESC**) **AS** Rank

**FROM** Order o, Time t,

**WHERE** o.timeID = t.timeID

**AND** t.Quarter = “Q1”

**AND TO\_CHAR**(SYSDATE, ‘YYYY’) = t.Year

**)**

**WHERE** Rank <=1;

**GROUP BY** agentKeY

**Data Warehouse design for a wholesale furniture company**

**1. Identify facts, dimensions and measures:**

Fact: Sales

Measures: Quantity, Income, Discount

Dimension: Furniture (Type, Category, Material)

Customer (Age, Gender,

**2. For each fact:**

 Produce the fact schema

**STAR SCHEMA**

**Diagram

Description automatically generated**

**a. Find the quantity, the total income and discount with respect to each city, type of furniture and the month**

SELECT SUM(quantity), SUM(income), SUM(discount), cityID, typeID, timeID

FROM Sales

GROUP BY cityID, typeID, timeID

**b. Find the average quantity, income and discount with respect to each country, furniture material and year**

SELECT AVG(quantity), AVG(income), AVG(discount), stateID, materialID, timeID

FROM Sales

GROUP BY stateID, materialID, timeID

**c. Determine the 5 most sold furniture during the May month**

SELECT typeID, SUM(quantity) as Total

FROM (

SELECT typeID, SUM(quantity) as Total,

RANK() OVER (ORDER BY SUM(quantity) DESC) as rank

FROM Sales s, Time t

WHERE t.month = “May”

)

WHERE rank <= 5