1 9 9 6

ATILIM

ÜNİVERSİTESİ

ECON485 Introduction to Database Systems

Term Project

Students: Sezin HOŞER 17232810038

Burcu YILMAZ 17232810028

Özden İlkim KURT 17232810033

Buket Nur KOCAKUŞAK 17232810003

Instructor: Bora GÜNGÖREN

FALL TERM, 2022

First, we installed the MySQL workbench and Intellij IDEA programs required for our project. Then we created a new model on the MySQL workbench that we downloaded.

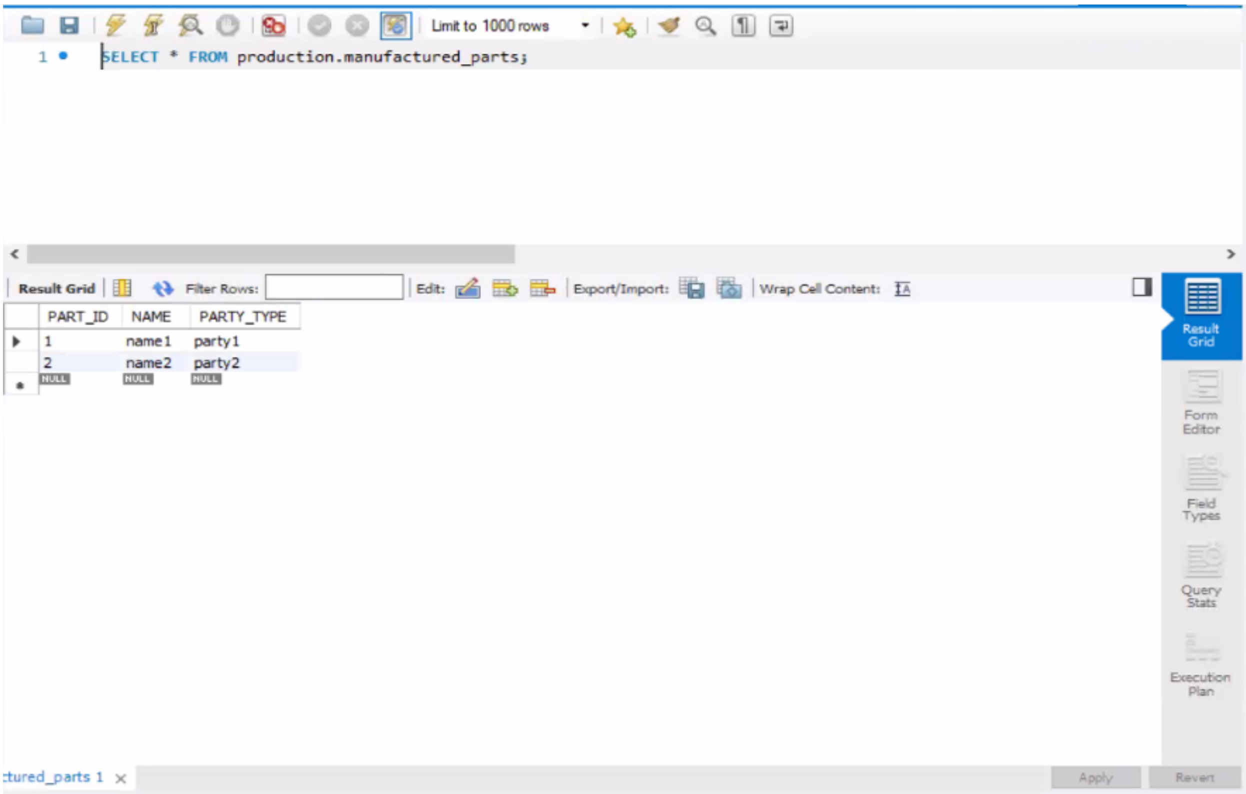

FIGURE 1: Manufactured_parts

We started to create the tables sequentially on the model we created. First, we created the manufactured_parts table, which consists of PART_ID, NAME and Party_TYPE columns, with the private key PART_ID.

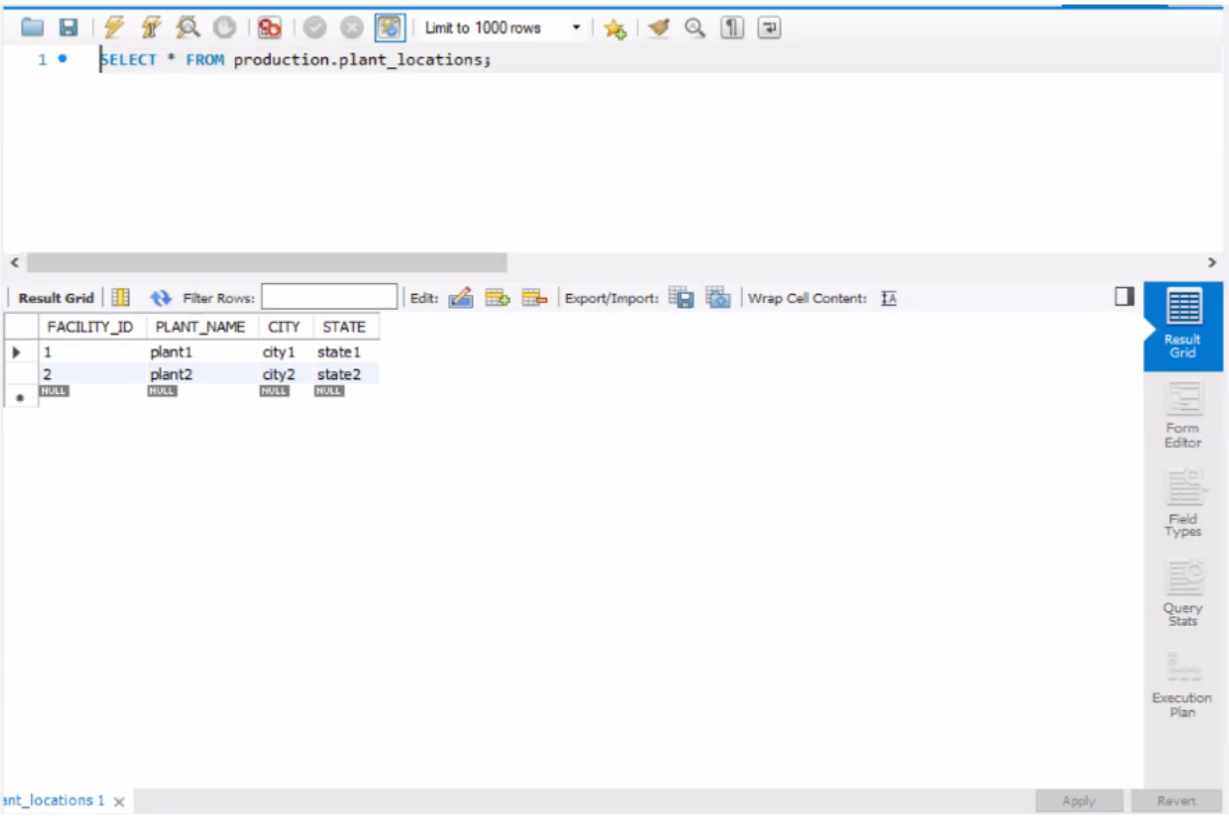

FIGURE 2: plant_locations

We added two pieces of data to the produced_parts table. We set the Private Keys of the added data to 1 and 2. Then we added the other tables respectively. We created the plan_locations table, which consists of FACILITY_ID, PLANT_NAME, CITY and STATE columns, with the private key FACILITY_ID.

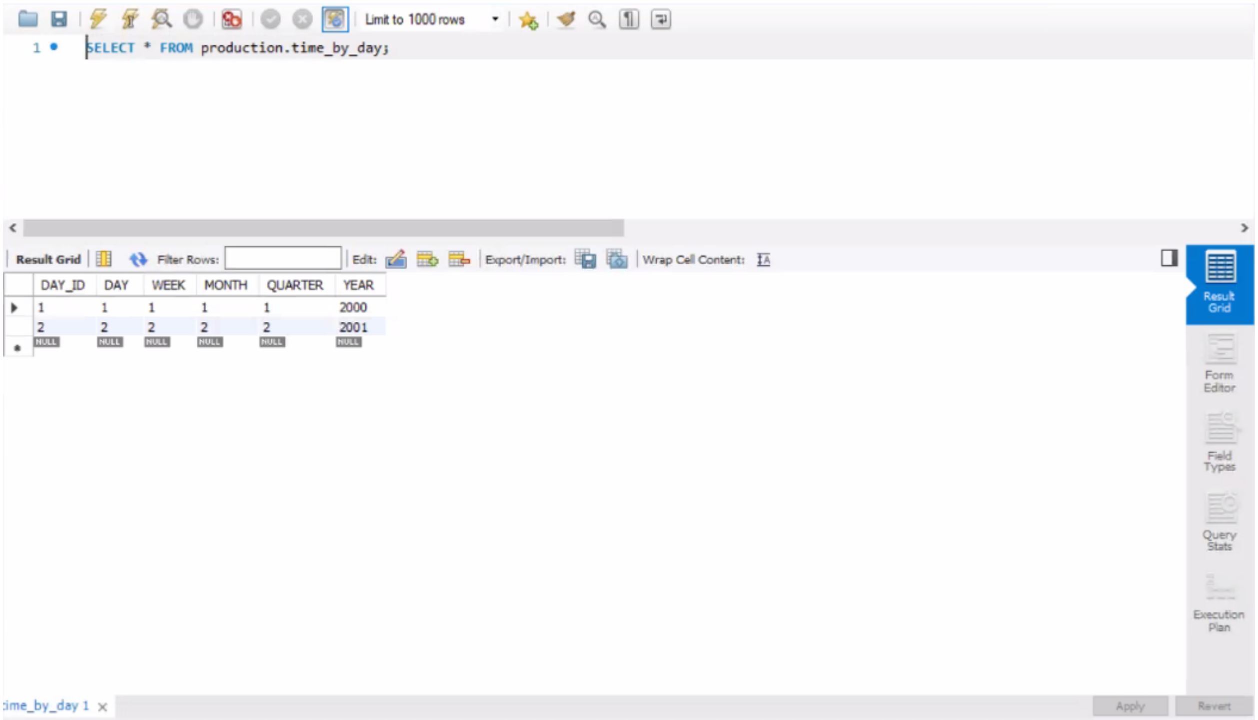

FIGURE 3: time_by_day

We created the time_by_day table consisting of DAY_ID, DAY, WEEK, MONTH, QUARTER and YEAR columns.

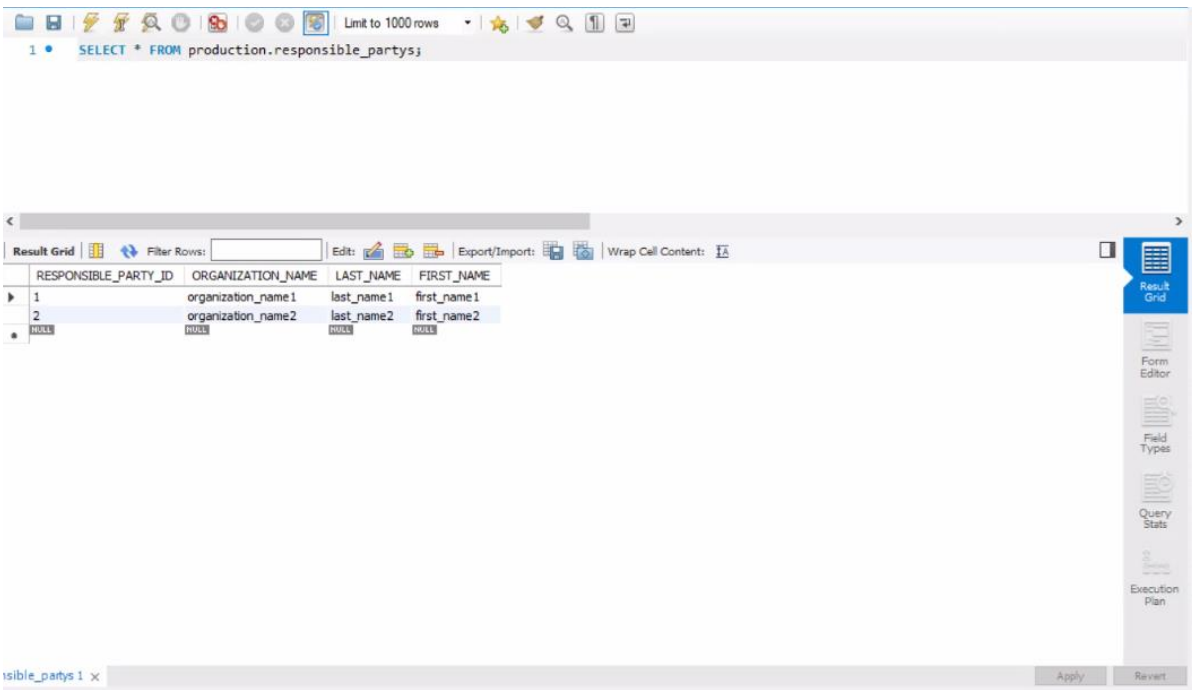

FIGURE 4: responsible_partys

We created the responsible_partys table, consisting of RESPONSIBLE_PARTY_ID, ORGANIZATION_NAME, LAST_NAME and FIRST_NAME columns.

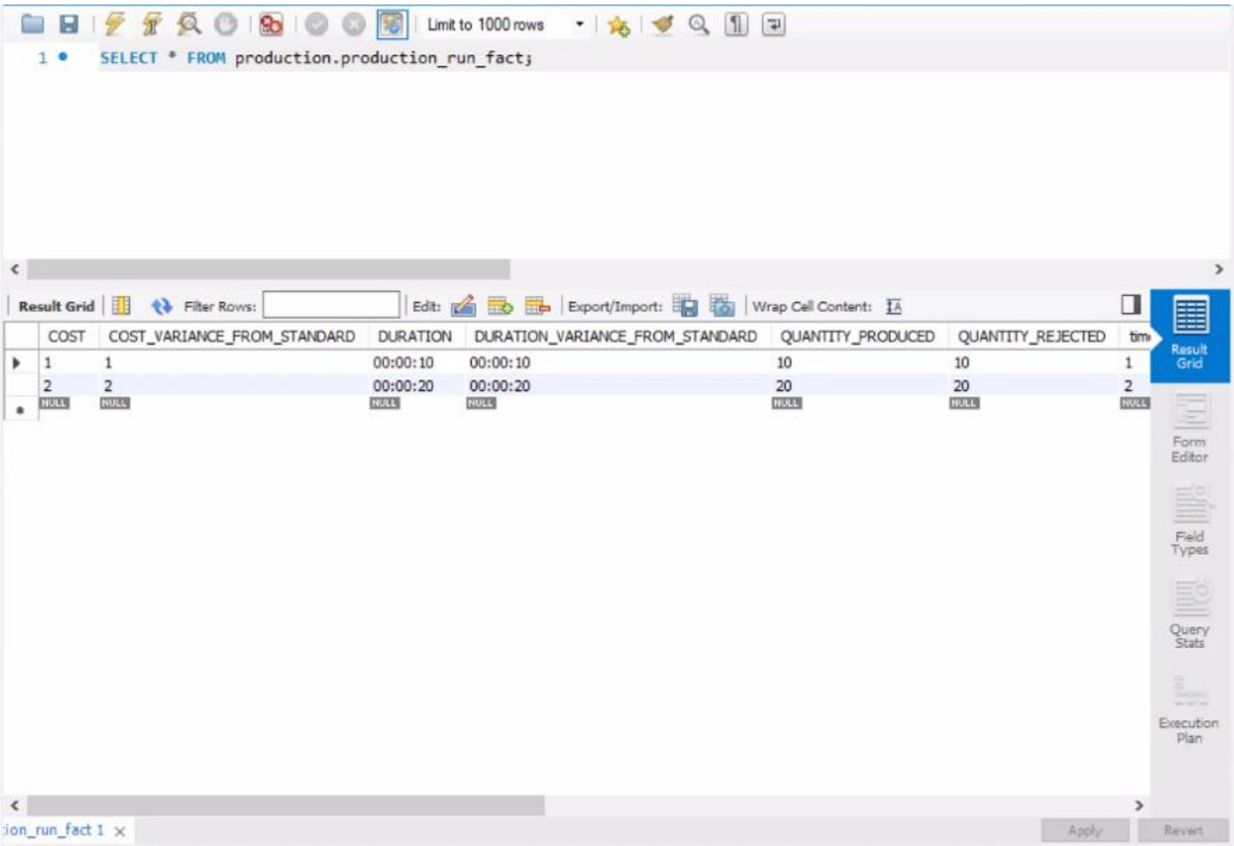

FIGURE 5: production_run_fact

We created the production_run_fact table with columns COST, COST_VARIANCE_FROM_STANDARD, DURATION, DURATION_VARIANCE_FROM_STANDARD, QUANTITY_PRODUCED and QUANTITY_REJECTED.

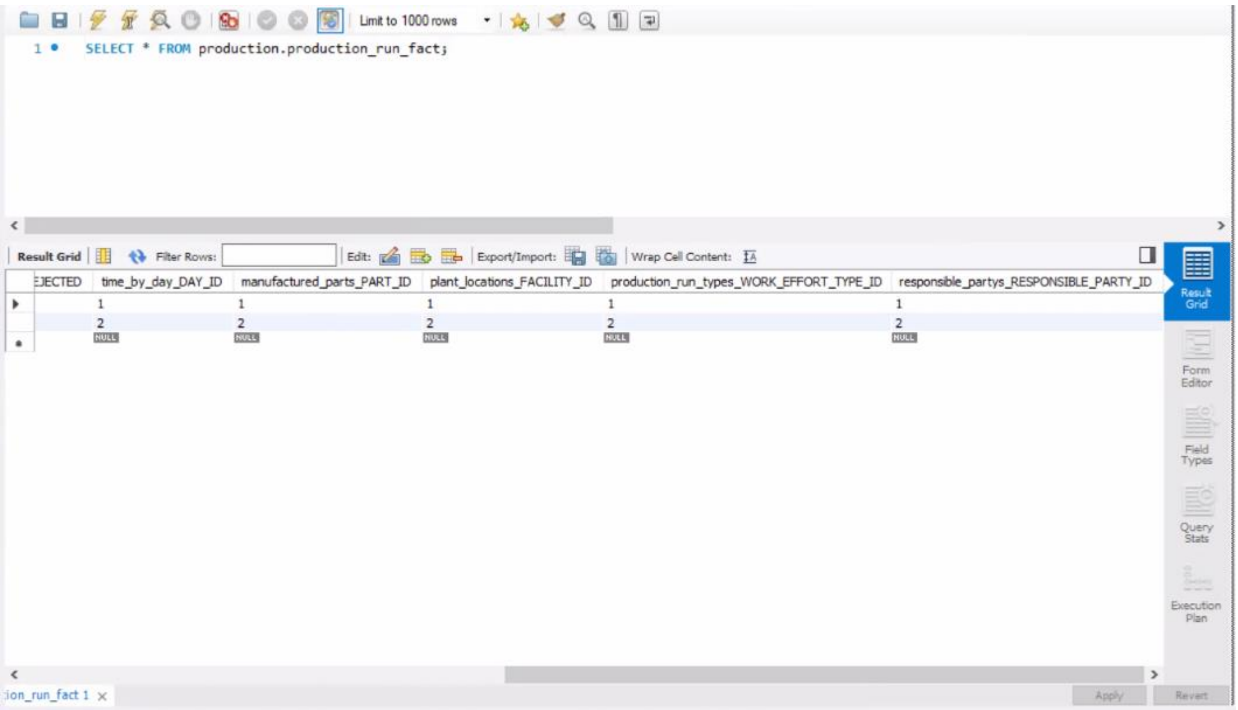

FIGURE 6: production_run_fact

Then, we linked the tables we created using EER Diagram and Private Keys.

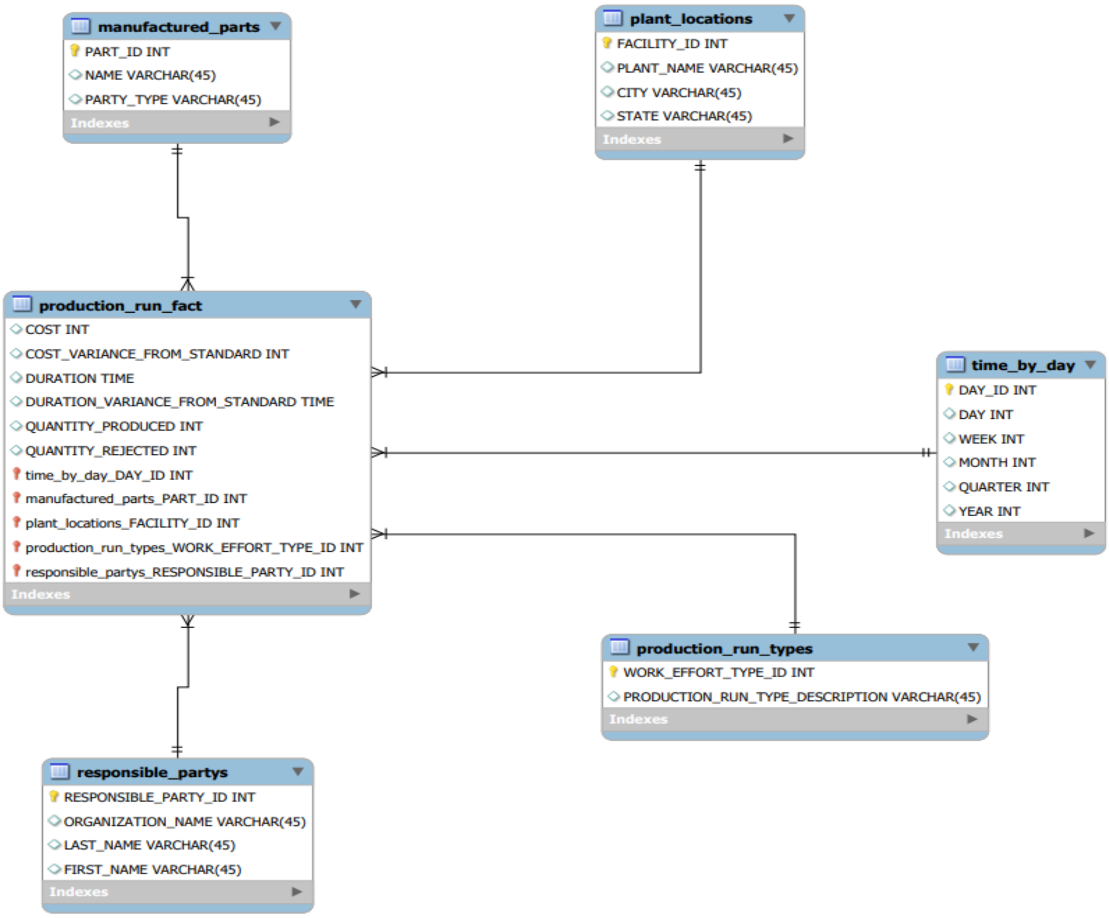

FIGURE 7: ERR Diagram

Production Run Fact The fact table, PRODUCTION_RUN_FACT, stores various measures for selected work efforts that have to do with production runs. The cost maintains how money was spent for the production runs that met the criteria. The cost variance_from_standard measures how much difference there was between the actual cost and the standard cost for the production runs (this may be extracted from the estimated cost attributes in the process plan EER diagram). The duration provides an average duration for work efforts and is based on the total time between the start and finish times of the work effort. This can be derived from the production run model. These start times and finish times are stored in the WORK EFFORT STATUS datetime for WORK EFFORT STATUS TYPES of “start" and "finished."

The duration variance from standard again measures the difference between the estimated duration from the process plan model and the actual duration (the previously described measure). The quantity produced comes directly from the quantity produced attribute in the PRODUCTION RUN, and the quantity rejected comes from the PRODUCTION RUN quantity rejected attribute, which is shown in ERR Diagram, quantity rejected attribute, which is shown in ERR Diagram. Dimensions Most of the data in this data mart design is sourced from information within the work efforts data models and specifically the subtype of WORK EFFORT named PRODUCTION_RUN. This PRODUCTION_RUN information is summarized into the dimensions MANUFACTURED_PARTS, PLANT_LOCATIONS, TIME_BY_DAY, PRODUCTION_RUN_TYPES, and RESPONSIBLE_PARTYS

The MANUFACTURED_PARTS information represents the part being manufactured and can be found by tracing the relationship from the WORK EFFORT to the corresponding PART being produced. The part_type is either "finished good" or "subassembly." If the enterprise needs it, parts can be further categorized. The PLANT_LOCATIONS dimension is sourced from the WORK EFFORT (PRODUCTION RUN) in this case, which is performed at a FACILITY (more specifically a PLANT), which in turn is located within a city and state using GEOGRAPHIC BOUNDARY. The facility_id will be sourced from the manufacturing facility information (FACILITY entity) where the product is produced, most likely a plant. This dimension allows operations managersious plants and analyze which ones are most productive.

The time dimension (TIME_BY_DAY) is granularized to the day and is selected based on the scheduled end date within the WORK EFFORT entity for subtype of PRODUCTION RUN. This could also be pulled from the scheduled start date of the WORK EFFORT entity (which is a supertype of PRO- DUCTION RUN). This dimension allows analysis of when the production runs are performed, indicating if there are any trends of productivity during various times of the year. The information is summarized to group together the statistics for PRODUC- TION RUN TYPESs (as opposed to showing individual production runs) because it is designed to analyze the efficiency of various types of production runs. The RESPONSIBLE_PARTYS entity shows which person or organization was responsible for the production run. This can be extracted from the WORK of compare var.

```
public static Connection getConn() throws  SQLException{
    Connection conn = null;

    String url       = "jdbc:mysql://localhost:3306/production";
    String user      = "root";
    String password  = "1234";

    conn = DriverManager.getConnection(url, user, password);
    return conn;
}
```

FIGURE 8: getConn() Function

When populating the tables, we first filled the manufactured_parts, plant_locations, time_by_day, production_run_types and responsible_partys tables. Because we are connecting these tables to the production_run_fact table, if we first filled the production_run_fact table, we would encounter an error by MySQL. After we created our tables on MySQL workbench I, we started to write our codes in java. In our code, we first wrote the getConn() function of the Connection type, which allows us to connect to the MySQL workbench. We entered the required Url, user and password information.

```java
    public static void main(String[] args) {
        try {
            Connection conn = null;
            conn = getConn();
            Statement statement = conn.createStatement();
            ResultSet resultSet = statement.executeQuery( sql: "select * from production_run_fact");
            System.out.println("PRODUCTION RUN FACT" + "\n");
            while (resultSet.next()){
                System.out.println(
                        "COST: " + resultSet.getString( columnLabel: "COST") + "\n" +
                        "COST_VARIANCE_FROM_STANDARD: " + resultSet.getString( columnLabel: "COST_VARIANCE_FROM_STANDARD") + "\n" +
                        "DURATION: " + resultSet.getString( columnLabel: "DURATION") + "\n" +
                        "DURATION_VARIENCE_FROM_STANDARD: " + resultSet.getString( columnLabel: "DURATION_VARIANCE_FROM_STANDARD") + "\n" +
                        "QUANTITY_PRODUCED: " + resultSet.getString( columnLabel: "QUANTITY_PRODUCED") + "\n" +
                        "QUANTITY_REJECTED: " + resultSet.getString( columnLabel: "QUANTITY_REJECTED") + "\n" +
                        "manufactoring_parts_PART_ID: " + resultSet.getString( columnLabel: "manufactured_parts_PART_ID") + "\n" +
                        "plant_locations_FACILITY_ID: " + resultSet.getString( columnLabel: "plant_locations_FACILITY_ID") + "\n" +
                        "time_by_day_DAY_ID: " + resultSet.getString( columnLabel: "time_by_day_DAY_ID") + "\n" +
                        "production_run_type_WORK_EFFORT_TYPE_ID: " + resultSet.getString( columnLabel: "production_run_types_WORK_EFFORT_TYPE_ID") + "\n" +
                        "responsible_partys_RESPONSIBLE_PARTY_ID: " + resultSet.getString( columnLabel: "responsible_partys_RESPONSIBLE_PARTY_ID") + "\n"
                );
            }

        } catch (SQLException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

FIGURE 9: Main

After providing the necessary connection, we wrote the necessary code to pull our data from MySQL.

```
"C:\Users\Burcu Yılmaz\.jdks\corretto-11.0.13\bin\java.exe" -agentlib:jdwp=transport=dt_socket,address=127.0.0.1:51410,suspend=y,server=n "-javaagent:C:\Users\BURCUY~1\AppData\L
Connected to the target VM, address: '127.0.0.1:51410', transport: 'socket'
PRODUCTION RUN FACT

COST: 1
COST_VARIANCE_FROM_STANDARD: 1
DURATION: 00:00:10
DURATION_VARIENCE_FROM_STANDARD: 00:00:10
QUANTITY_PRODUCED: 10
QUANTITY_REJECTED: 10
manufactoring_parts_PART_ID: 1
plant_locations_FACILITY_ID: 1
time_by_day_DAY_ID: 1
production_run_type_WORK_EFFORT_TYPE_ID: 1
responsible_partys_RESPONSIBLE_PARTY_ID: 1

COST: 2
COST_VARIANCE_FROM_STANDARD: 2
DURATION: 00:00:20
DURATION_VARIENCE_FROM_STANDARD: 00:00:20
QUANTITY_PRODUCED: 20
QUANTITY_REJECTED: 20
manufactoring_parts_PART_ID: 2
plant_locations_FACILITY_ID: 2
time_by_day_DAY_ID: 2
production_run_type_WORK_EFFORT_TYPE_ID: 2
responsible_partys_RESPONSIBLE_PARTY_ID: 2

Disconnected from the target VM, address: '127.0.0.1:51410', transport: 'socket'
```

FIGURE 10: Results

For example, we pulled our data from the database according to the column names from the production_run_fack table. Then we printed our outputs with System.out.println.

Enterprise Resource Planning (ERP)

Manufacturing ERP refers to Enterprise Resource Planning (ERP) software and systems used to plan, manage and deliver specific functions that support manufacturers and manufacturing business operations. Modern manufacturing ERP systems are built to flexibly support and integrate any business process, creating a complete business management platform for manufacturing companies. ERP systems are a type of production management software that increases the organizational efficiency of a manufacturing business by managing and improving how company resources are used. Increasing and/or reducing the number of resources required without sacrificing quality and performance is key to effectively improving manufacturing business growth and profitability. With ERP software, manufacturing companies have the ability to manage critical aspects of everything from shop floor operations to procurement and inventory planning. There are many valuable operational and financial benefits to having a modern, integrated manufacturing ERP system at both the micro and macro level. In this project we have done, we normally need to fill the tables with ERP software, but in the code in our project, we only read the tables with select. This is mostly done by industrial engineers, we just filled in the tables manually in this project.