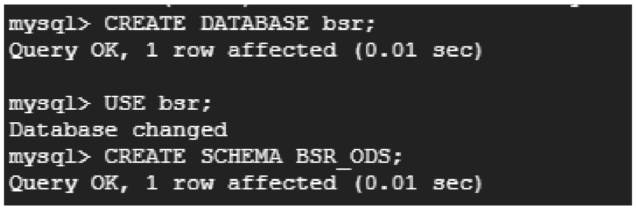
DSBA 6160 Final Report

Harley Grafton
Joe Burns
Priyanka Kamath

Create Database & Schema

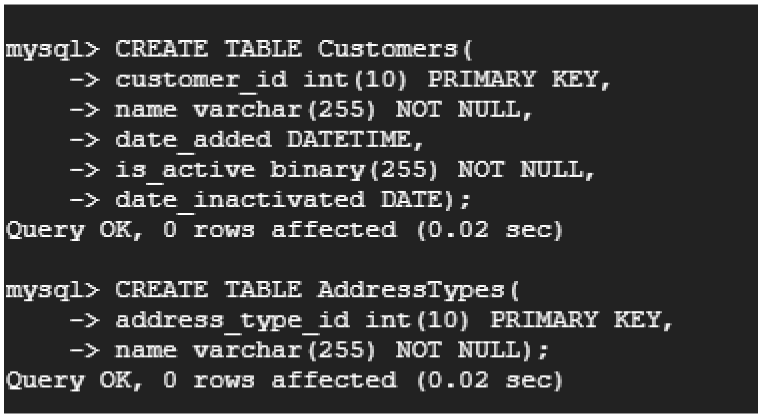


```
mysql> CREATE DATABASE bsr;
Query OK, 1 row affected (0.01 sec)

mysql> USE bsr;
Database changed
mysql> CREATE SCHEMA BSR_ODS;
Query OK, 1 row affected (0.01 sec)
```

Creating Tables

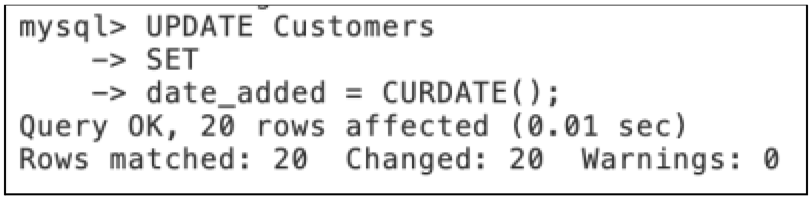
<u>ODS Schema</u>

Customers (20 customers)



```
mysql> CREATE TABLE Customers(
    -> customer_id int(10) PRIMARY KEY,
    -> name varchar(255) NOT NULL,
    -> date_added DATETIME,
    -> is_active binary(255) NOT NULL,
    -> date_inactivated DATE);
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE TABLE AddressTypes(
    -> address_type_id int(10) PRIMARY KEY,
    -> name varchar(255) NOT NULL);
Query OK, 0 rows affected (0.02 sec)
```

Changed the date format for date_added.

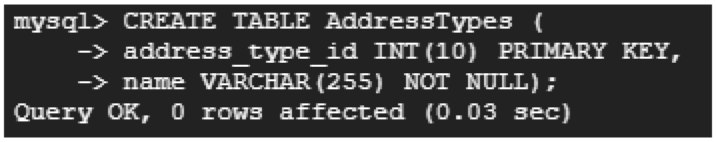


```
mysql> UPDATE Customers
    -> SET
    -> date_added = CURDATE();
Query OK, 20 rows affected (0.01 sec)
Rows matched: 20  Changed: 20  Warnings: 0
```

Updated is_active data type from binary(255) to BOOLEAN.

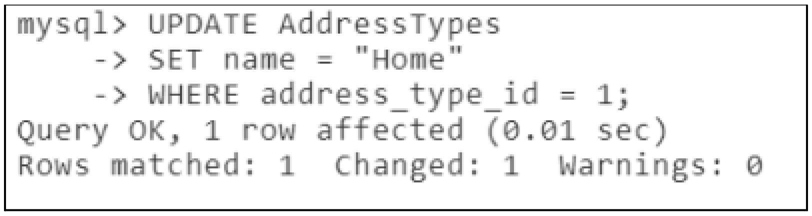
date_added was set to NOT NULL, screenshot not available.

AddressTypes (3 address types)

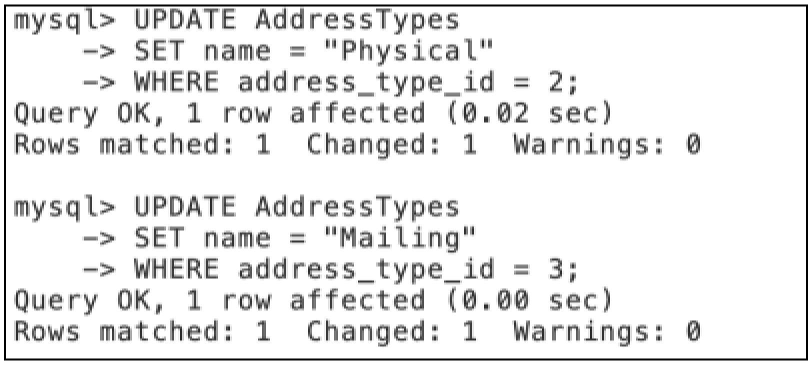


```
mysql> CREATE TABLE AddressTypes (
    -> address_type_id INT(10) PRIMARY KEY,
    -> name VARCHAR(255) NOT NULL);
Query OK, 0 rows affected (0.03 sec)
```

Updated AddressTypes and set names for each address_type_id.

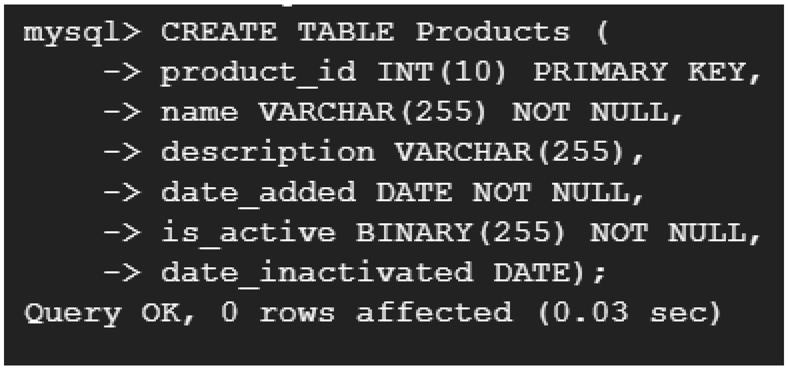


```
mysql> UPDATE AddressTypes
    -> SET name = "Home"
    -> WHERE address_type_id = 1;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```



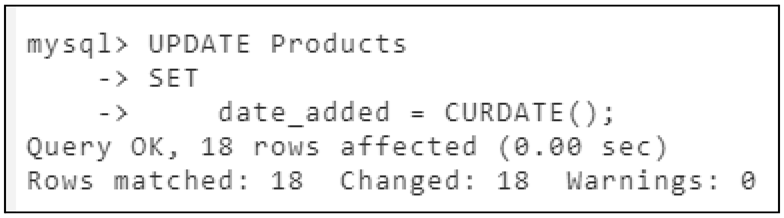
```
mysql> UPDATE AddressTypes
    -> SET name = "Physical"
    -> WHERE address_type_id = 2;
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> UPDATE AddressTypes
    -> SET name = "Mailing"
    -> WHERE address_type_id = 3;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Products (18 products)



```
mysql> CREATE TABLE Products (
    -> product_id INT(10) PRIMARY KEY,
    -> name VARCHAR(255) NOT NULL,
    -> description VARCHAR(255),
    -> date_added DATE NOT NULL,
    -> is_active BINARY(255) NOT NULL,
    -> date_inactivated DATE);
Query OK, 0 rows affected (0.03 sec)
```

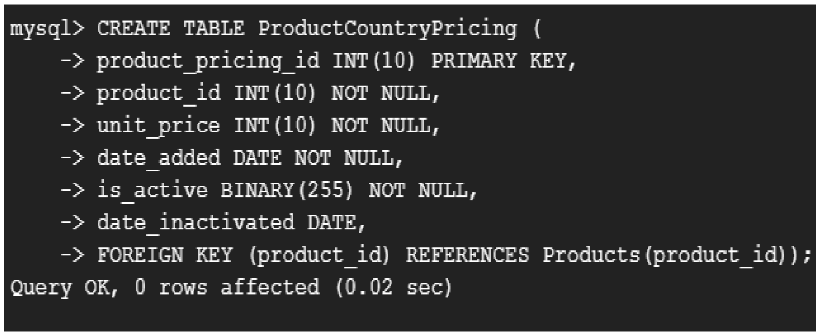
Updated date_added data type from DATE to CURDATE().



```
mysql> UPDATE Products
    -> SET
    ->      date_added = CURDATE();
Query OK, 18 rows affected (0.00 sec)
Rows matched: 18  Changed: 18  Warnings: 0
```

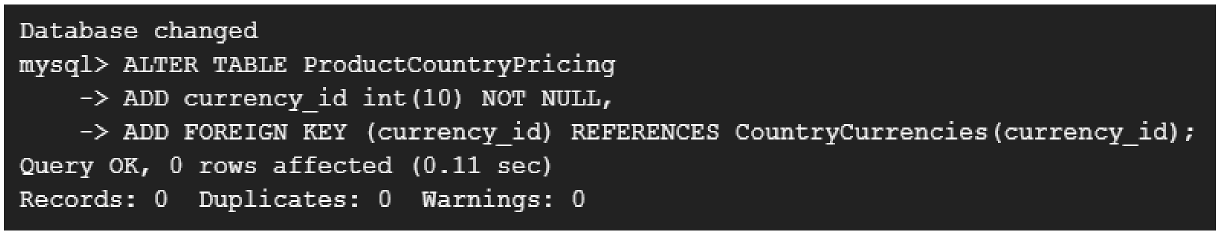
Updated is_active data type from binary(255) to BOOLEAN, screenshot not available.

ProductCountryPricing



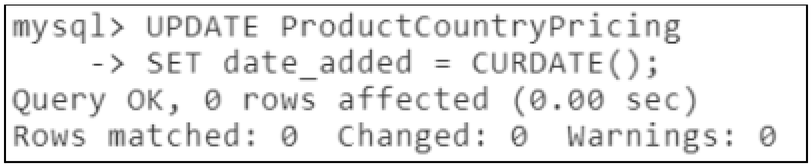
```
mysql> CREATE TABLE ProductCountryPricing (
    -> product_pricing_id INT(10) PRIMARY KEY,
    -> product_id INT(10) NOT NULL,
    -> unit_price INT(10) NOT NULL,
    -> date_added DATE NOT NULL,
    -> is_active BINARY(255) NOT NULL,
    -> date_inactivated DATE,
    -> FOREIGN KEY (product_id) REFERENCES Products(product_id));
Query OK, 0 rows affected (0.02 sec)
```

Addition of currency_id and foreign key reference once Country Currencies table was created.



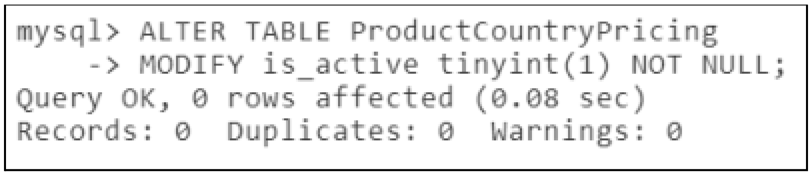
```
Database changed
mysql> ALTER TABLE ProductCountryPricing
    -> ADD currency_id int(10) NOT NULL,
    -> ADD FOREIGN KEY (currency_id) REFERENCES CountryCurrencies(currency_id);
Query OK, 0 rows affected (0.11 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Updated date_added data type from DATE to CURDATE().



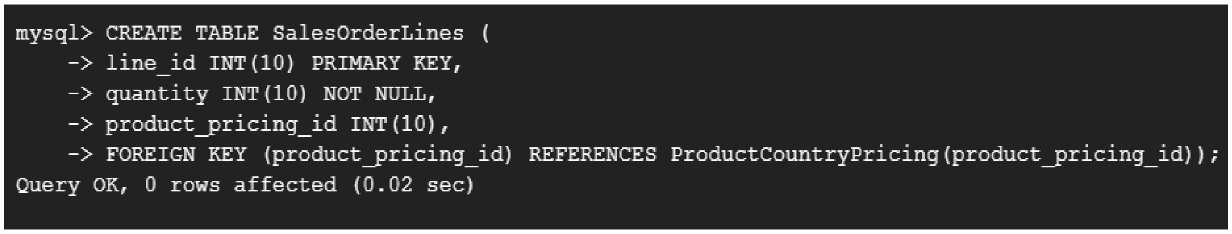
```
mysql> UPDATE ProductCountryPricing
    -> SET date_added = CURDATE();
Query OK, 0 rows affected (0.00 sec)
Rows matched: 0  Changed: 0  Warnings: 0
```

Updated is_active data type from binary(255) to BOOLEAN.



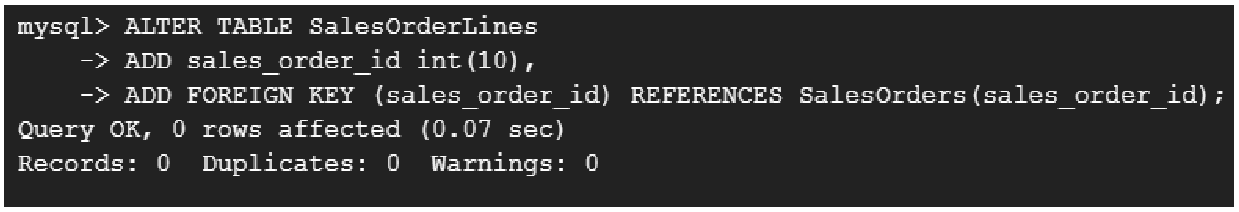
```
mysql> ALTER TABLE ProductCountryPricing
    -> MODIFY is_active tinyint(1) NOT NULL;
Query OK, 0 rows affected (0.08 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

SalesOrderLines (50 sales order lines)



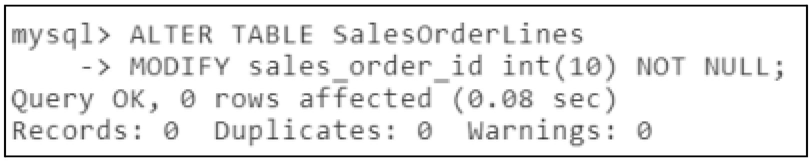
```
mysql> CREATE TABLE SalesOrderLines (
    -> line_id INT(10) PRIMARY KEY,
    -> quantity INT(10) NOT NULL,
    -> product_pricing_id INT(10),
    -> FOREIGN KEY (product_pricing_id) REFERENCES ProductCountryPricing(product_pricing_id));
Query OK, 0 rows affected (0.02 sec)
```

Addition of the sales_order_id foreign and foreign key reference once the SalesOrders table was created.



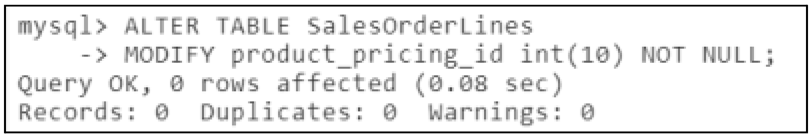
```
mysql> ALTER TABLE SalesOrderLines
    -> ADD sales_order_id int(10),
    -> ADD FOREIGN KEY (sales_order_id) REFERENCES SalesOrders(sales_order_id);
Query OK, 0 rows affected (0.07 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Update sales_order_id to NOT NULL.



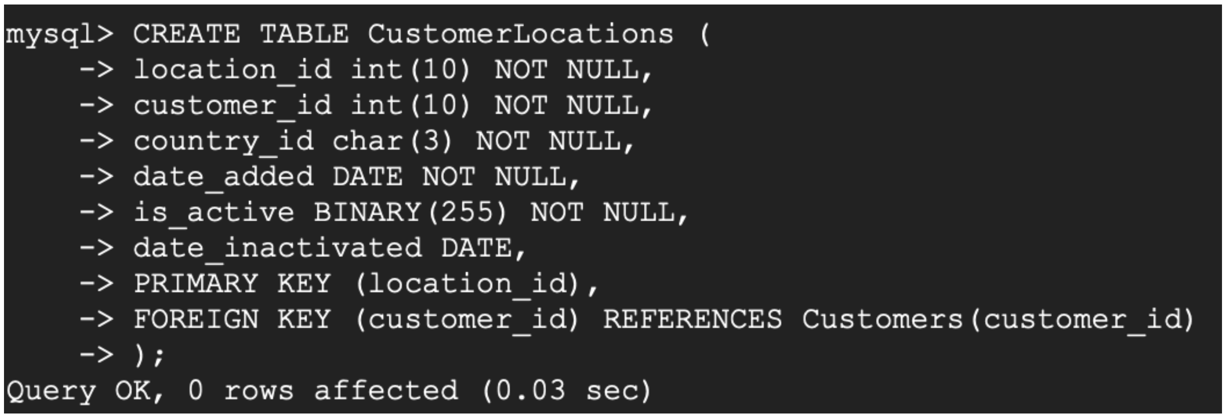
```
mysql> ALTER TABLE SalesOrderLines
    -> MODIFY sales_order_id int(10) NOT NULL;
Query OK, 0 rows affected (0.08 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Update product_pricing_id to NOT NULL.



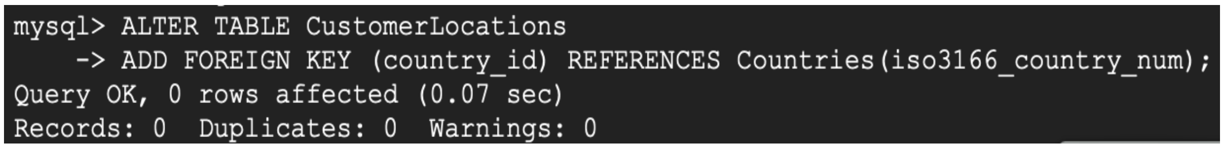
```
mysql> ALTER TABLE SalesOrderLines
    -> MODIFY product_pricing_id int(10) NOT NULL;
Query OK, 0 rows affected (0.08 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

CustomerLocations



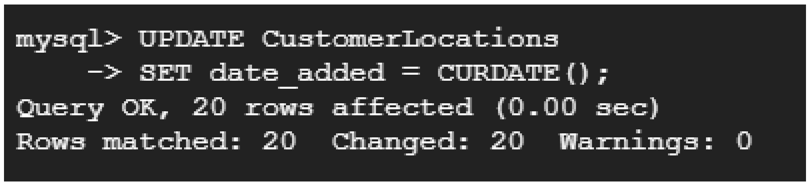
```
mysql> CREATE TABLE CustomerLocations (
    -> location_id int(10) NOT NULL,
    -> customer_id int(10) NOT NULL,
    -> country_id char(3) NOT NULL,
    -> date_added DATE NOT NULL,
    -> is_active BINARY(255) NOT NULL,
    -> date_inactivated DATE,
    -> PRIMARY KEY (location_id),
    -> FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)
    -> );
Query OK, 0 rows affected (0.03 sec)
```

Addition of the country_id foreign key reference once the Countries table was created.



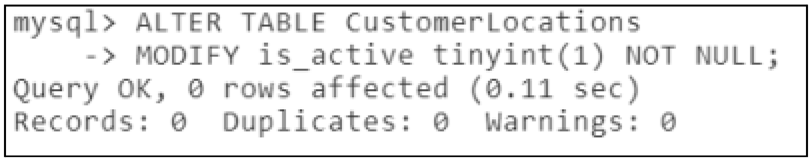
```
mysql> ALTER TABLE CustomerLocations
    -> ADD FOREIGN KEY (country_id) REFERENCES Countries(iso3166_country_num);
Query OK, 0 rows affected (0.07 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Changed the date format for date_added.



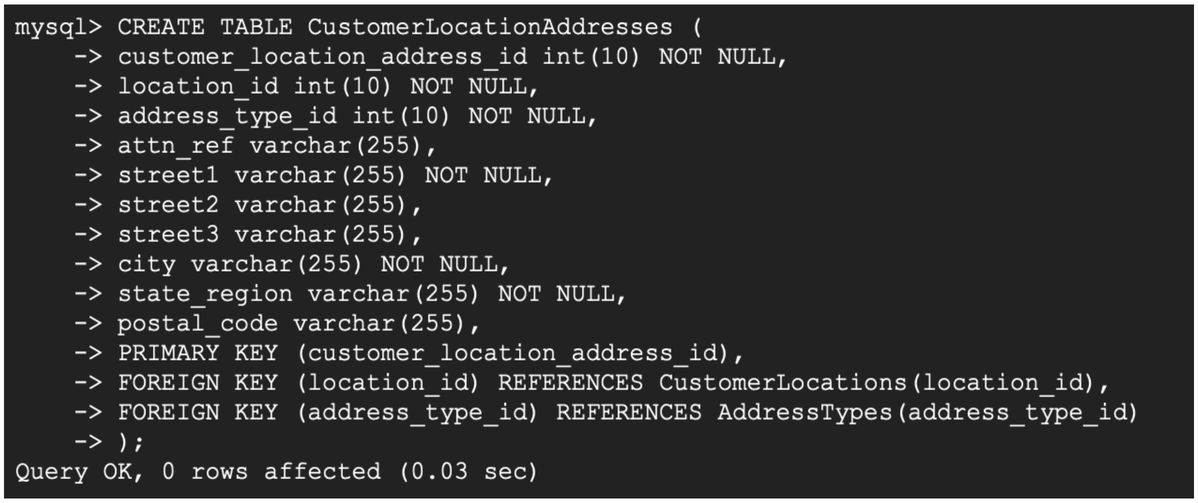
```
mysql> UPDATE CustomerLocations
    -> SET date_added = CURDATE();
Query OK, 20 rows affected (0.00 sec)
Rows matched: 20  Changed: 20  Warnings: 0
```

Updated is_active data type from binary(255) to BOOLEAN.



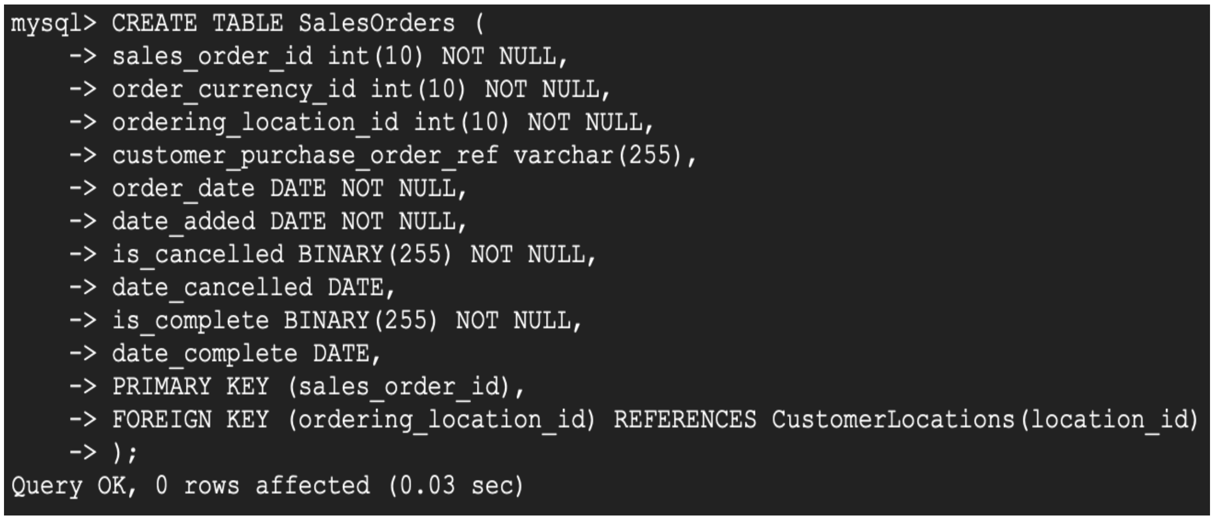
```
mysql> ALTER TABLE CustomerLocations
    -> MODIFY is_active tinyint(1) NOT NULL;
Query OK, 0 rows affected (0.11 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

CustomerLocationAddresses



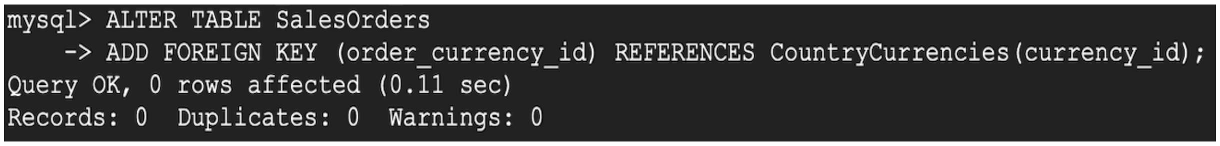
```
mysql> CREATE TABLE CustomerLocationAddresses (
    -> customer_location_address_id int(10) NOT NULL,
    -> location_id int(10) NOT NULL,
    -> address_type_id int(10) NOT NULL,
    -> attn_ref varchar(255),
    -> street1 varchar(255) NOT NULL,
    -> street2 varchar(255),
    -> street3 varchar(255),
    -> city varchar(255) NOT NULL,
    -> state_region varchar(255) NOT NULL,
    -> postal_code varchar(255),
    -> PRIMARY KEY (customer_location_address_id),
    -> FOREIGN KEY (location_id) REFERENCES CustomerLocations(location_id),
    -> FOREIGN KEY (address_type_id) REFERENCES AddressTypes(address_type_id)
    -> );
Query OK, 0 rows affected (0.03 sec)
```

SalesOrders (50 sales orders)



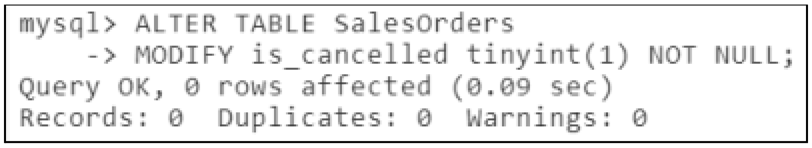
```
mysql> CREATE TABLE SalesOrders (
    -> sales_order_id int(10) NOT NULL,
    -> order_currency_id int(10) NOT NULL,
    -> ordering_location_id int(10) NOT NULL,
    -> customer_purchase_order_ref varchar(255),
    -> order_date DATE NOT NULL,
    -> date_added DATE NOT NULL,
    -> is_cancelled BINARY(255) NOT NULL,
    -> date_cancelled DATE,
    -> is_complete BINARY(255) NOT NULL,
    -> date_complete DATE,
    -> PRIMARY KEY (sales_order_id),
    -> FOREIGN KEY (ordering_location_id) REFERENCES CustomerLocations(location_id)
    -> );
Query OK, 0 rows affected (0.03 sec)
```

Addition of the order_currency_id foreign key reference once the CountryCurrencies table was created.



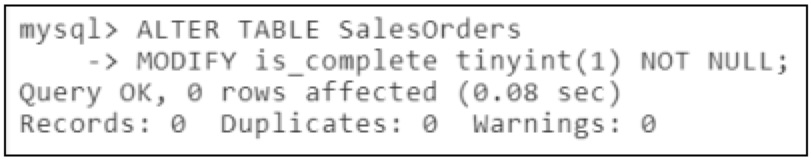
```
mysql> ALTER TABLE SalesOrders
    -> ADD FOREIGN KEY (order_currency_id) REFERENCES CountryCurrencies(currency_id);
Query OK, 0 rows affected (0.11 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Updated is_cancelled data type from binary(255) to BOOLEAN.



```
mysql> ALTER TABLE SalesOrders
    -> MODIFY is_cancelled tinyint(1) NOT NULL;
Query OK, 0 rows affected (0.09 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

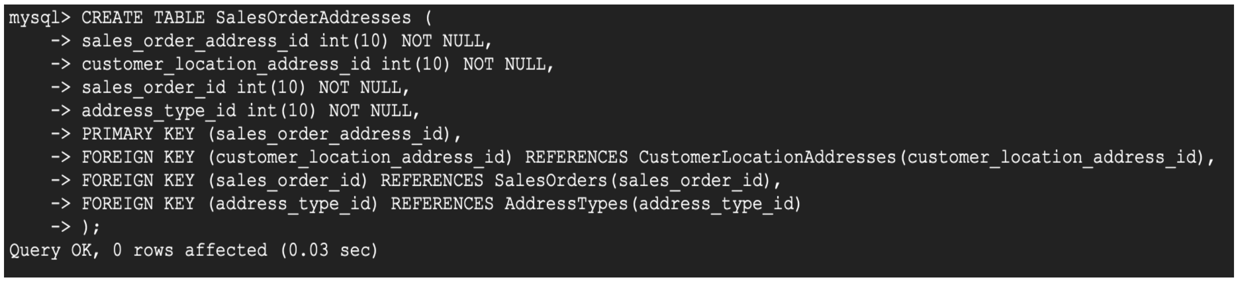
Updated is_complete data type from binary(255) to BOOLEAN.



```
mysql> ALTER TABLE SalesOrders
    -> MODIFY is_complete tinyint(1) NOT NULL;
Query OK, 0 rows affected (0.08 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

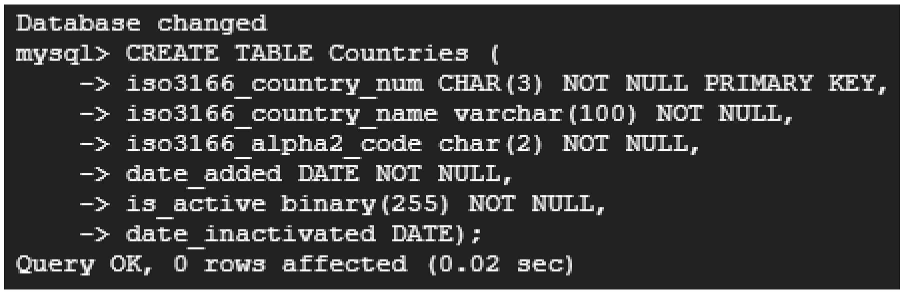
customer_purchase_order_ref was set to NOT NULL.

SalesOrderAddresses



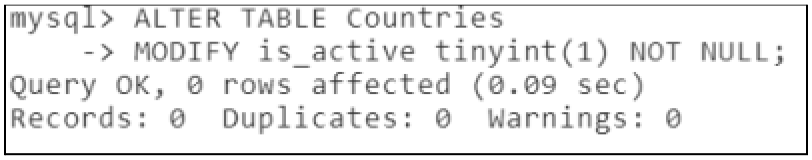
```
mysql> CREATE TABLE SalesOrderAddresses (
    -> sales_order_address_id int(10) NOT NULL,
    -> customer_location_address_id int(10) NOT NULL,
    -> sales_order_id int(10) NOT NULL,
    -> address_type_id int(10) NOT NULL,
    -> PRIMARY KEY (sales_order_address_id),
    -> FOREIGN KEY (customer_location_address_id) REFERENCES CustomerLocationAddresses(customer_location_address_id),
    -> FOREIGN KEY (sales_order_id) REFERENCES SalesOrders(sales_order_id),
    -> FOREIGN KEY (address_type_id) REFERENCES AddressTypes(address_type_id)
    -> );
Query OK, 0 rows affected (0.03 sec)
```

Countries



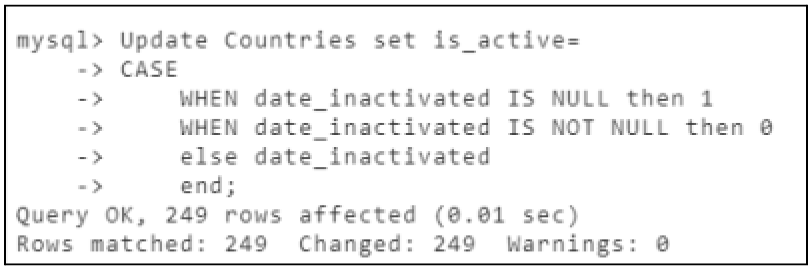
```
Database changed
mysql> CREATE TABLE Countries (
    -> iso3166_country_num CHAR(3) NOT NULL PRIMARY KEY,
    -> iso3166_country_name varchar(100) NOT NULL,
    -> iso3166_alpha2_code char(2) NOT NULL,
    -> date_added DATE NOT NULL,
    -> is_active binary(255) NOT NULL,
    -> date_inactivated DATE);
Query OK, 0 rows affected (0.02 sec)
```

Updated is_active data type from binary(255) to BOOLEAN.



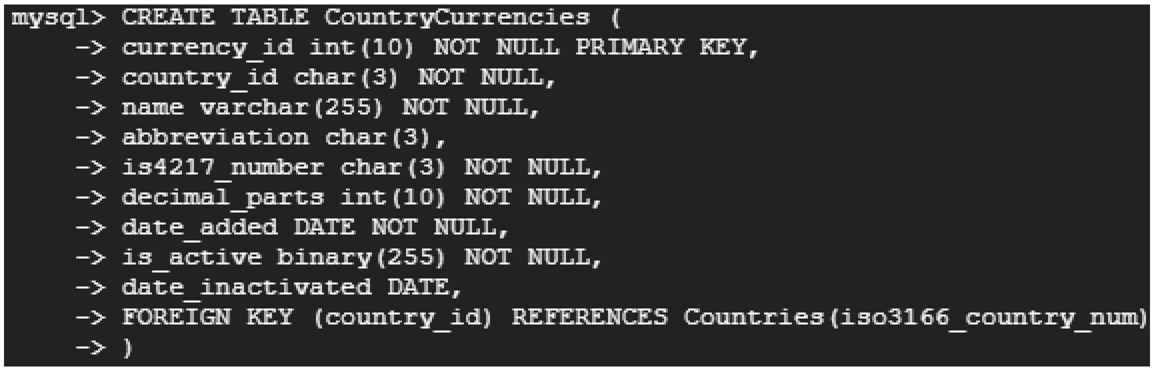
```
mysql> ALTER TABLE Countries
    -> MODIFY is_active tinyint(1) NOT NULL;
Query OK, 0 rows affected (0.09 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Updated date_inactivated values.



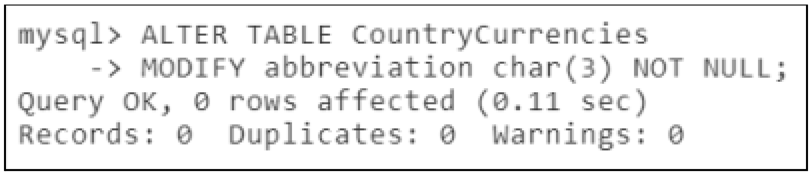
```
mysql> Update Countries set is_active=
    -> CASE
    ->     WHEN date_inactivated IS NULL then 1
    ->     WHEN date_inactivated IS NOT NULL then 0
    ->     else date_inactivated
    ->     end;
Query OK, 249 rows affected (0.01 sec)
Rows matched: 249  Changed: 249  Warnings: 0
```

CountryCurrencies



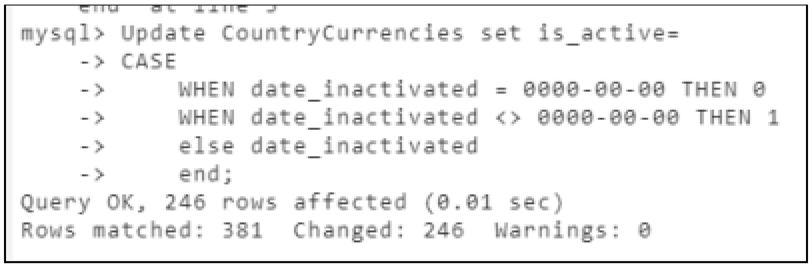
```
mysql> CREATE TABLE CountryCurrencies (
    -> currency_id int(10) NOT NULL PRIMARY KEY,
    -> country_id char(3) NOT NULL,
    -> name varchar(255) NOT NULL,
    -> abbreviation char(3),
    -> is4217_number char(3) NOT NULL,
    -> decimal_parts int(10) NOT NULL,
    -> date_added DATE NOT NULL,
    -> is_active binary(255) NOT NULL,
    -> date_inactivated DATE,
    -> FOREIGN KEY (country_id) REFERENCES Countries(iso3166_country_num)
    -> )
```

Abbreviation set to NOT NULL.



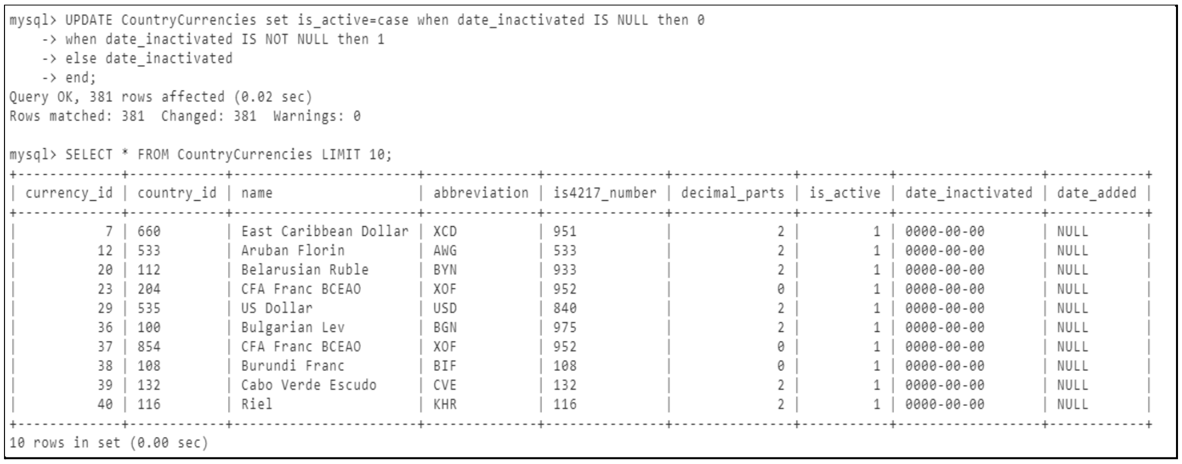
```
mysql> ALTER TABLE CountryCurrencies
    -> MODIFY abbreviation char(3) NOT NULL;
Query OK, 0 rows affected (0.11 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Updated is_active data type from binary(255) to BOOLEAN.



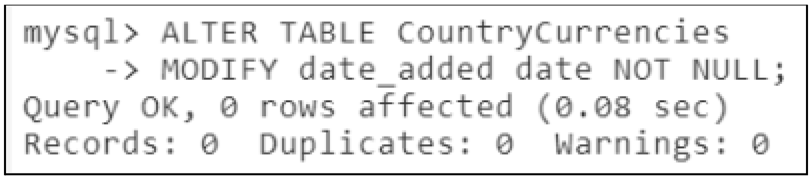
```
mysql> Update CountryCurrencies set is_active=
    -> CASE
    ->     WHEN date_inactivated = 0000-00-00 THEN 0
    ->     WHEN date_inactivated <> 0000-00-00 THEN 1
    ->     else date_inactivated
    ->     end;
Query OK, 246 rows affected (0.01 sec)
Rows matched: 381  Changed: 246  Warnings: 0
```

Updated is_active values.



```
mysql> UPDATE CountryCurrencies set is_active=case when date_inactivated IS NULL then 0
    -> when date_inactivated IS NOT NULL then 1
    -> else date_inactivated
    -> end;
Query OK, 381 rows affected (0.02 sec)
Rows matched: 381  Changed: 381  Warnings: 0

mysql> SELECT * FROM CountryCurrencies LIMIT 10;
+-------------+------------+----------------------+--------------+---------------+---------------+-----------+------------------+------------+
| currency_id | country_id | name                 | abbreviation | is4217_number | decimal_parts | is_active | date_inactivated | date_added |
+-------------+------------+----------------------+--------------+---------------+---------------+-----------+------------------+------------+
|           7 | 660        | East Caribbean Dollar | XCD         | 951           |             2 |         1 | 0000-00-00       | NULL       |
|          12 | 533        | Aruban Florin        | AWG          | 533           |             2 |         1 | 0000-00-00       | NULL       |
|          20 | 112        | Belarusian Ruble     | BYN          | 933           |             2 |         1 | 0000-00-00       | NULL       |
|          23 | 204        | CFA Franc BCEAO      | XOF          | 952           |             0 |         1 | 0000-00-00       | NULL       |
|          29 | 535        | US Dollar            | USD          | 840           |             2 |         1 | 0000-00-00       | NULL       |
|          36 | 100        | Bulgarian Lev        | BGN          | 975           |             2 |         1 | 0000-00-00       | NULL       |
|          37 | 854        | CFA Franc BCEAO      | XOF          | 952           |             0 |         1 | 0000-00-00       | NULL       |
|          38 | 108        | Burundi Franc        | BIF          | 108           |             0 |         1 | 0000-00-00       | NULL       |
|          39 | 132        | Cabo Verde Escudo    | CVE          | 132           |             2 |         1 | 0000-00-00       | NULL       |
|          40 | 116        | Riel                 | KHR          | 116           |             2 |         1 | 0000-00-00       | NULL       |
+-------------+------------+----------------------+--------------+---------------+---------------+-----------+------------------+------------+
10 rows in set (0.00 sec)
```

Modified date_added to NOT NULL.

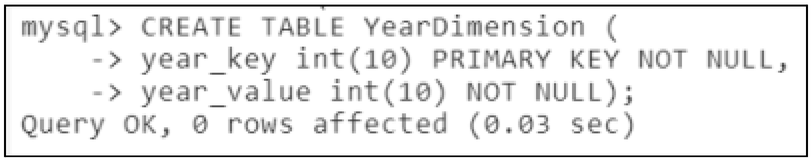


```
mysql> ALTER TABLE CountryCurrencies
    -> MODIFY date_added date NOT NULL;
Query OK, 0 rows affected (0.08 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Updated date_added data type from DATE to CURDATE().

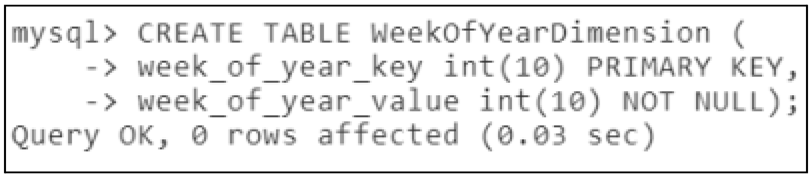
DW Schema

YearDimension



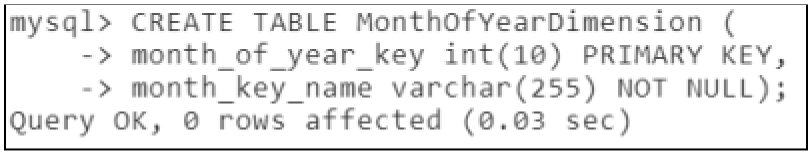
```
mysql> CREATE TABLE YearDimension (
    -> year_key int(10) PRIMARY KEY NOT NULL,
    -> year_value int(10) NOT NULL);
Query OK, 0 rows affected (0.03 sec)
```

WeekOfYearDimension



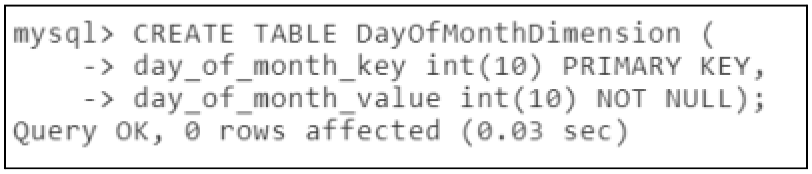
```
mysql> CREATE TABLE WeekOfYearDimension (
    -> week_of_year_key int(10) PRIMARY KEY,
    -> week_of_year_value int(10) NOT NULL);
Query OK, 0 rows affected (0.03 sec)
```

MonthOfYearDimension



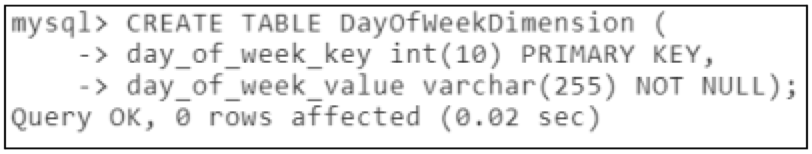
```
mysql> CREATE TABLE MonthOfYearDimension (
    -> month_of_year_key int(10) PRIMARY KEY,
    -> month_key_name varchar(255) NOT NULL);
Query OK, 0 rows affected (0.03 sec)
```

DayOfMonthDimension



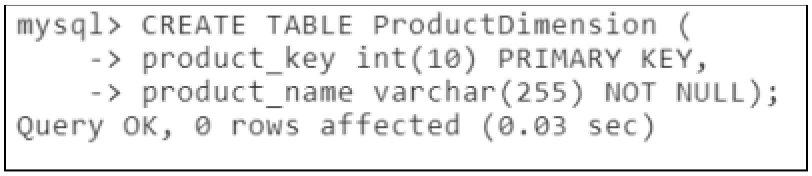
```
mysql> CREATE TABLE DayOfMonthDimension (
    -> day_of_month_key int(10) PRIMARY KEY,
    -> day_of_month_value int(10) NOT NULL);
Query OK, 0 rows affected (0.03 sec)
```

DayOfWeekDimension



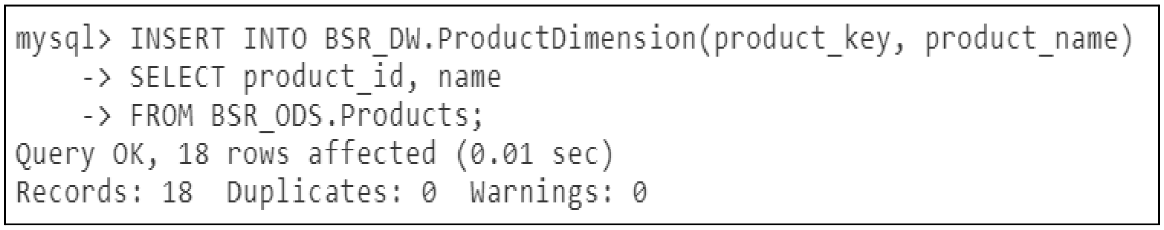
```
mysql> CREATE TABLE DayOfWeekDimension (
    -> day_of_week_key int(10) PRIMARY KEY,
    -> day_of_week_value varchar(255) NOT NULL);
Query OK, 0 rows affected (0.02 sec)
```

ProductDimension



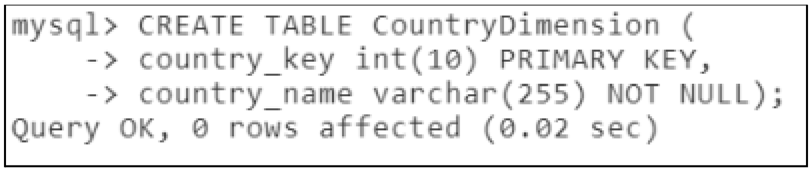
```
mysql> CREATE TABLE ProductDimension (
    -> product_key int(10) PRIMARY KEY,
    -> product_name varchar(255) NOT NULL);
Query OK, 0 rows affected (0.03 sec)
```

Insert product_id from BSR_ODS Products table into product_key from the BSR_DW ProductDimensions table.



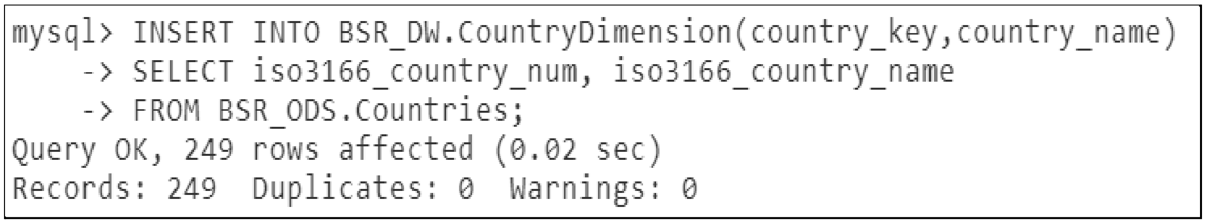
```
mysql> INSERT INTO BSR_DW.ProductDimension(product_key, product_name)
    -> SELECT product_id, name
    -> FROM BSR_ODS.Products;
Query OK, 18 rows affected (0.01 sec)
Records: 18  Duplicates: 0  Warnings: 0
```

CountryDimension



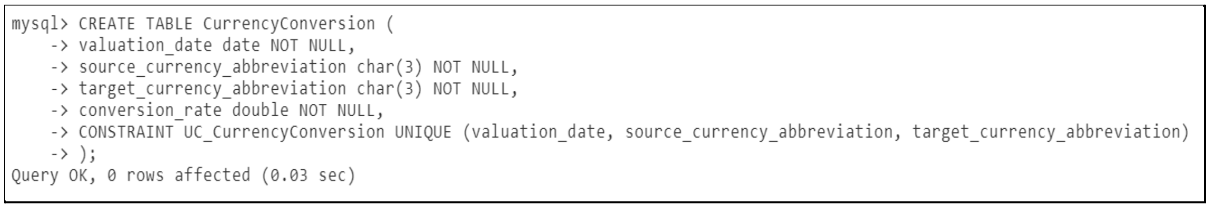
```
mysql> CREATE TABLE CountryDimension (
    -> country_key int(10) PRIMARY KEY,
    -> country_name varchar(255) NOT NULL);
Query OK, 0 rows affected (0.02 sec)
```

Insert iso number and associated country name from BSR_ODS Countries table into country_key and country_name from the BSR_DW CountryDimension table.



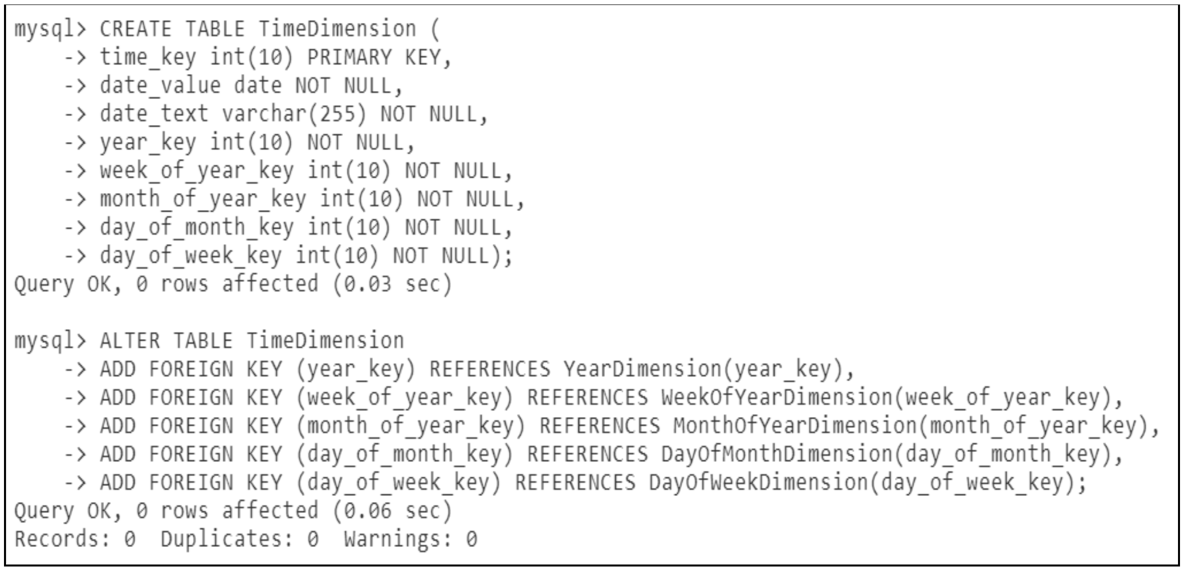
```
mysql> INSERT INTO BSR_DW.CountryDimension(country_key,country_name)
    -> SELECT iso3166_country_num, iso3166_country_name
    -> FROM BSR_ODS.Countries;
Query OK, 249 rows affected (0.02 sec)
Records: 249  Duplicates: 0  Warnings: 0
```

CurrencyConversion



```
mysql> CREATE TABLE CurrencyConversion (
    -> valuation_date date NOT NULL,
    -> source_currency_abbreviation char(3) NOT NULL,
    -> target_currency_abbreviation char(3) NOT NULL,
    -> conversion_rate double NOT NULL,
    -> CONSTRAINT UC_CurrencyConversion UNIQUE (valuation_date, source_currency_abbreviation, target_currency_abbreviation)
    -> );
Query OK, 0 rows affected (0.03 sec)
```

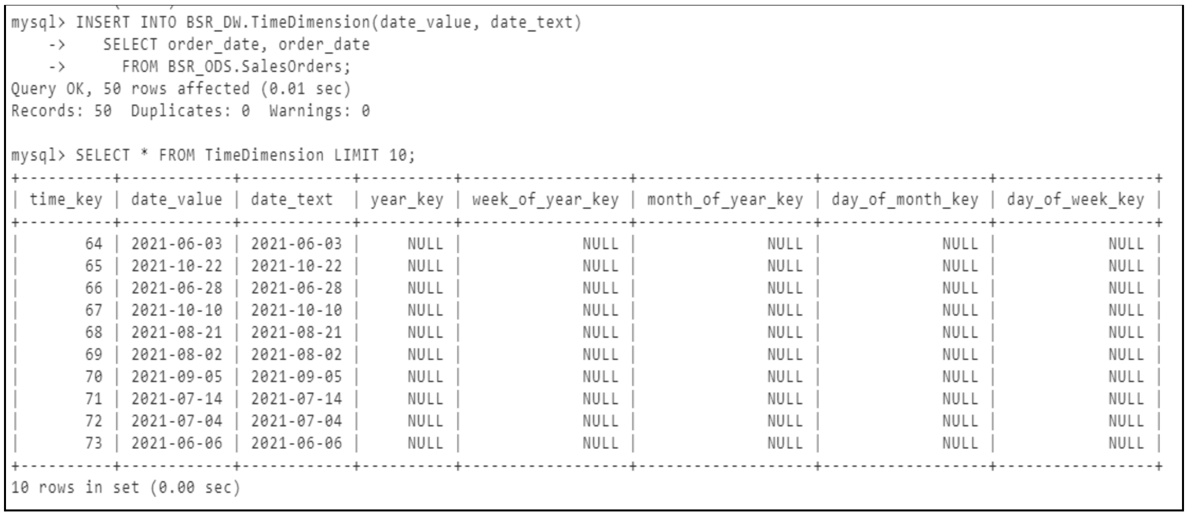
TimeDimension



```
mysql> CREATE TABLE TimeDimension (
    -> time_key int(10) PRIMARY KEY,
    -> date_value date NOT NULL,
    -> date_text varchar(255) NOT NULL,
    -> year_key int(10) NOT NULL,
    -> week_of_year_key int(10) NOT NULL,
    -> month_of_year_key int(10) NOT NULL,
    -> day_of_month_key int(10) NOT NULL,
    -> day_of_week_key int(10) NOT NULL);
Query OK, 0 rows affected (0.03 sec)

mysql> ALTER TABLE TimeDimension
    -> ADD FOREIGN KEY (year_key) REFERENCES YearDimension(year_key),
    -> ADD FOREIGN KEY (week_of_year_key) REFERENCES WeekOfYearDimension(week_of_year_key),
    -> ADD FOREIGN KEY (month_of_year_key) REFERENCES MonthOfYearDimension(month_of_year_key),
    -> ADD FOREIGN KEY (day_of_month_key) REFERENCES DayOfMonthDimension(day_of_month_key),
    -> ADD FOREIGN KEY (day_of_week_key) REFERENCES DayOfWeekDimension(day_of_week_key);
Query OK, 0 rows affected (0.06 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

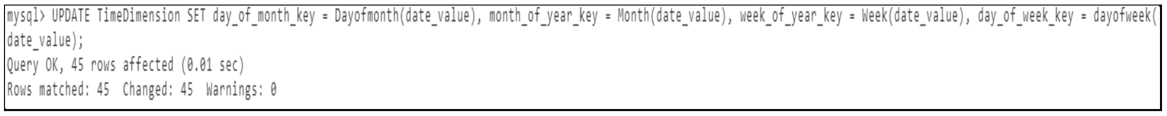
Inserting ordar_date from BSR_ODS SalesOrder table into date_value and date_text in the DSR_DW Time Dimension table.

This was later re-done as SELECT DISTINCT rather than just SELECT. No screenshot available though. Result was 45 distinct order dates, with 5 dates appearing twice.



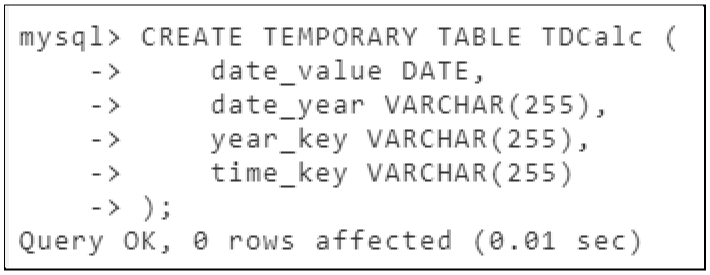
```
mysql> INSERT INTO BSR_DW.TimeDimension(date_value, date_text)
    ->    SELECT order_date, order_date
    ->      FROM BSR_ODS.SalesOrders;
Query OK, 50 rows affected (0.01 sec)
Records: 50  Duplicates: 0  Warnings: 0

mysql> SELECT * FROM TimeDimension LIMIT 10;
+----------+------------+------------+----------+------------------+-------------------+------------------+-----------------+
| time_key | date_value | date_text  | year_key | week_of_year_key | month_of_year_key | day_of_month_key | day_of_week_key |
+----------+------------+------------+----------+------------------+-------------------+------------------+-----------------+
|       64 | 2021-06-03 | 2021-06-03 |     NULL |             NULL |              NULL |             NULL |            NULL |
|       65 | 2021-10-22 | 2021-10-22 |     NULL |             NULL |              NULL |             NULL |            NULL |
|       66 | 2021-06-28 | 2021-06-28 |     NULL |             NULL |              NULL |             NULL |            NULL |
|       67 | 2021-10-10 | 2021-10-10 |     NULL |             NULL |              NULL |             NULL |            NULL |
|       68 | 2021-08-21 | 2021-08-21 |     NULL |             NULL |              NULL |             NULL |            NULL |
|       69 | 2021-08-02 | 2021-08-02 |     NULL |             NULL |              NULL |             NULL |            NULL |
|       70 | 2021-09-05 | 2021-09-05 |     NULL |             NULL |              NULL |             NULL |            NULL |
|       71 | 2021-07-14 | 2021-07-14 |     NULL |             NULL |              NULL |             NULL |            NULL |
|       72 | 2021-07-04 | 2021-07-04 |     NULL |             NULL |              NULL |             NULL |            NULL |
|       73 | 2021-06-06 | 2021-06-06 |     NULL |             NULL |              NULL |             NULL |            NULL |
+----------+------------+------------+----------+------------------+-------------------+------------------+-----------------+
10 rows in set (0.00 sec)
```

Inserting date_value from BSR_ODS SalesOrder table into day_of_month_key, month_of_year_key, week_of_year_key, and day_of_week_key in the BSR_DW TimeDimension table.

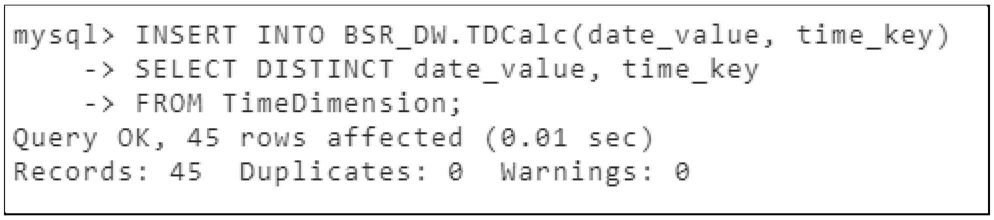


```
mysql> UPDATE TimeDimension SET day_of_month_key = Dayofmonth(date_value), month_of_year_key = Month(date_value), week_of_year_key = Week(date_value), day_of_week_key = dayofweek(
date_value);
Query OK, 45 rows affected (0.01 sec)
Rows matched: 45  Changed: 45  Warnings: 0
```

Creating Temporary Table TDCalc to be used for year_key calculation.

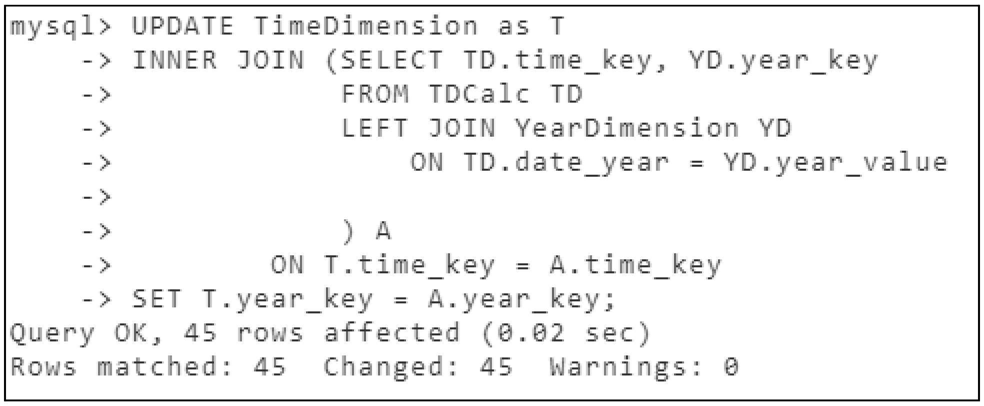


```
mysql> CREATE TEMPORARY TABLE TDCalc (
    ->     date_value DATE,
    ->     date_year VARCHAR(255),
    ->     year_key VARCHAR(255),
    ->     time_key VARCHAR(255)
    -> );
Query OK, 0 rows affected (0.01 sec)
```



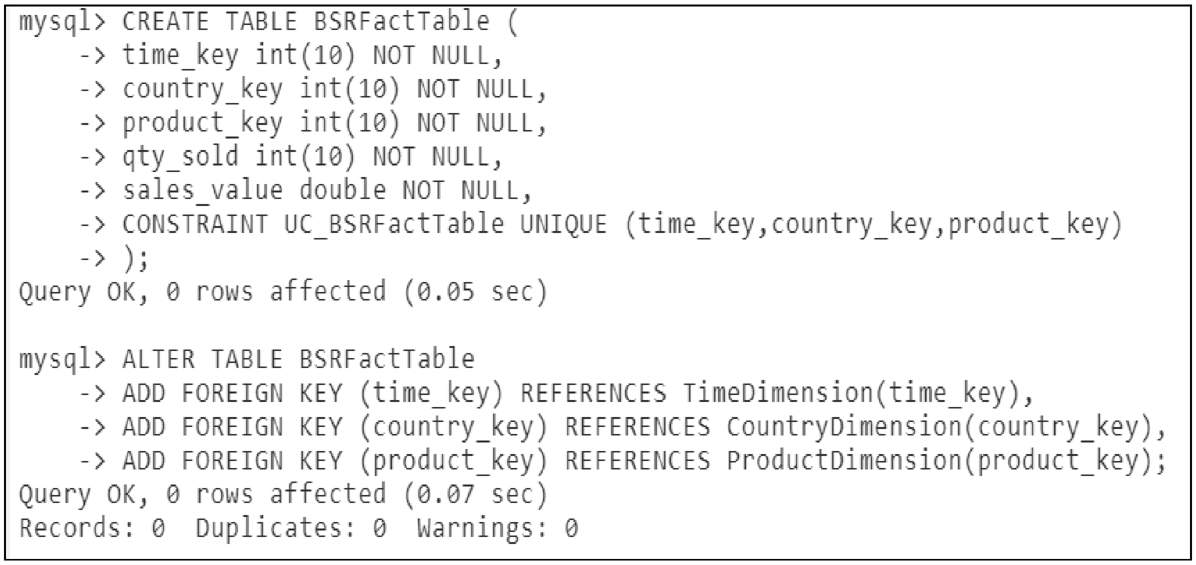
```
mysql> INSERT INTO BSR_DW.TDCalc(date_value, time_key)
    -> SELECT DISTINCT date_value, time_key
    -> FROM TimeDimension;
Query OK, 45 rows affected (0.01 sec)
Records: 45  Duplicates: 0  Warnings: 0
```

Loading year_key through subquery to left join year to year_key with dimensional table.



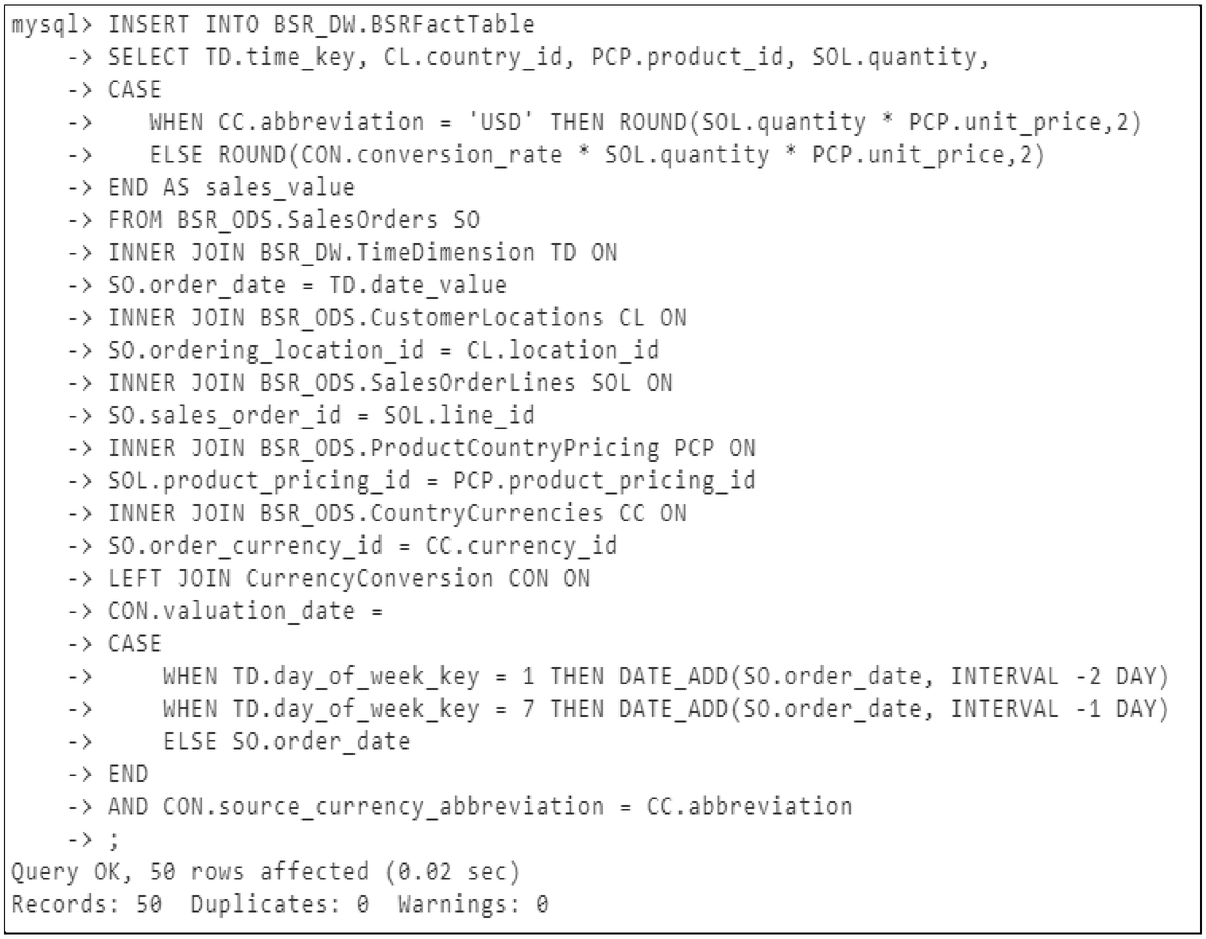
```
mysql> UPDATE TimeDimension as T
    -> INNER JOIN (SELECT TD.time_key, YD.year_key
    ->             FROM TDCalc TD
    ->             LEFT JOIN YearDimension YD
    ->                 ON TD.date_year = YD.year_value
    ->
    ->             ) A
    ->         ON T.time_key = A.time_key
    -> SET T.year_key = A.year_key;
Query OK, 45 rows affected (0.02 sec)
Rows matched: 45  Changed: 45  Warnings: 0
```

BSRFactTable

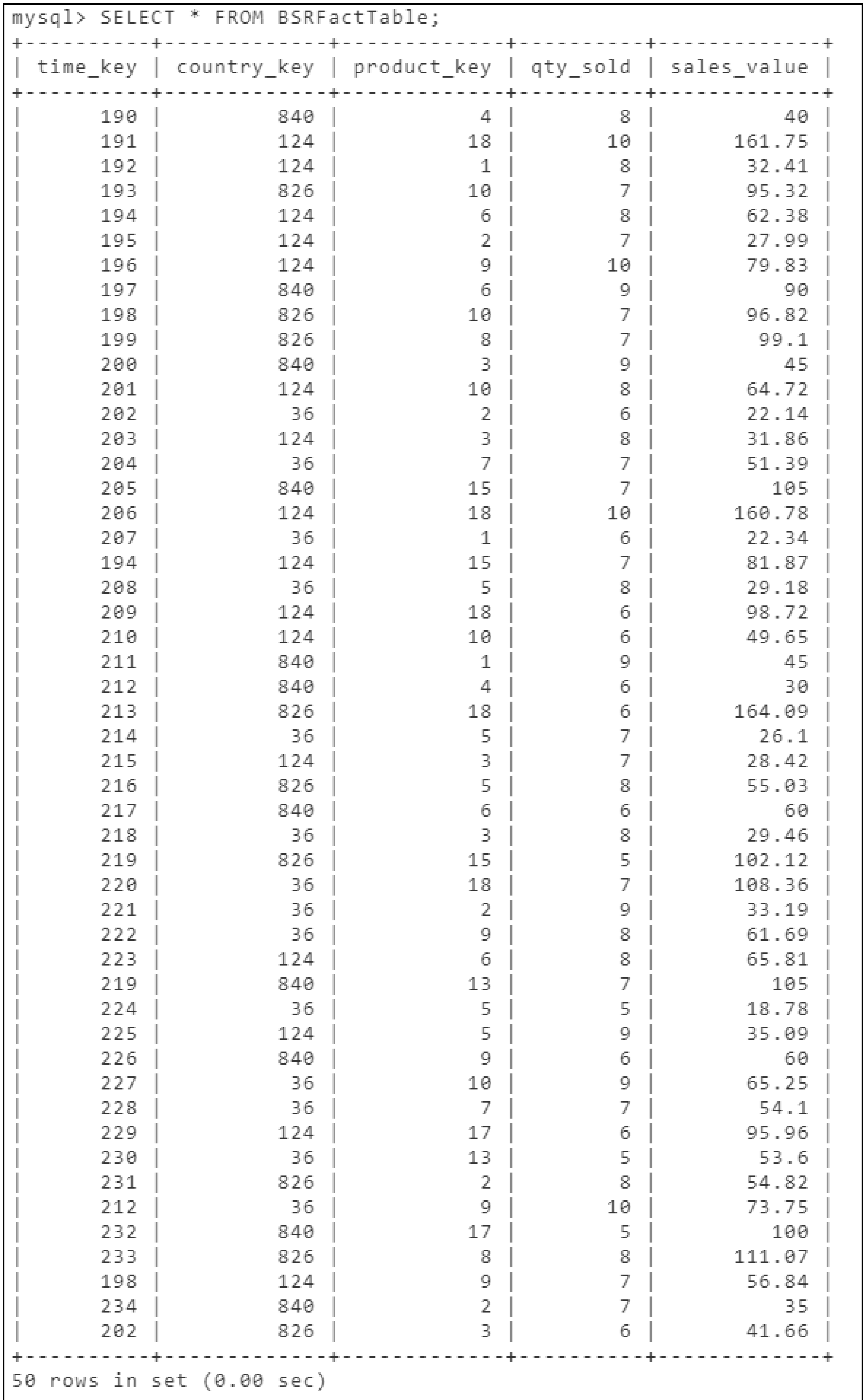


```
mysql> CREATE TABLE BSRFactTable (
    -> time_key int(10) NOT NULL,
    -> country_key int(10) NOT NULL,
    -> product_key int(10) NOT NULL,
    -> qty_sold int(10) NOT NULL,
    -> sales_value double NOT NULL,
    -> CONSTRAINT UC_BSRFactTable UNIQUE (time_key,country_key,product_key)
    -> );
Query OK, 0 rows affected (0.05 sec)

mysql> ALTER TABLE BSRFactTable
    -> ADD FOREIGN KEY (time_key) REFERENCES TimeDimension(time_key),
    -> ADD FOREIGN KEY (country_key) REFERENCES CountryDimension(country_key),
    -> ADD FOREIGN KEY (product_key) REFERENCES ProductDimension(product_key);
Query OK, 0 rows affected (0.07 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Loading query results.



```
mysql> INSERT INTO BSR_DW.BSRFactTable
    -> SELECT TD.time_key, CL.country_id, PCP.product_id, SOL.quantity,
    -> CASE
    ->    WHEN CC.abbreviation = 'USD' THEN ROUND(SOL.quantity * PCP.unit_price,2)
    ->    ELSE ROUND(CON.conversion_rate * SOL.quantity * PCP.unit_price,2)
    -> END AS sales_value
    -> FROM BSR_ODS.SalesOrders SO
    -> INNER JOIN BSR_DW.TimeDimension TD ON
    -> SO.order_date = TD.date_value
    -> INNER JOIN BSR_ODS.CustomerLocations CL ON
    -> SO.ordering_location_id = CL.location_id
    -> INNER JOIN BSR_ODS.SalesOrderLines SOL ON
    -> SO.sales_order_id = SOL.line_id
    -> INNER JOIN BSR_ODS.ProductCountryPricing PCP ON
    -> SOL.product_pricing_id = PCP.product_pricing_id
    -> INNER JOIN BSR_ODS.CountryCurrencies CC ON
    -> SO.order_currency_id = CC.currency_id
    -> LEFT JOIN CurrencyConversion CON ON
    -> CON.valuation_date =
    -> CASE
    ->     WHEN TD.day_of_week_key = 1 THEN DATE_ADD(SO.order_date, INTERVAL -2 DAY)
    ->     WHEN TD.day_of_week_key = 7 THEN DATE_ADD(SO.order_date, INTERVAL -1 DAY)
    ->     ELSE SO.order_date
    -> END
    -> AND CON.source_currency_abbreviation = CC.abbreviation
    -> ;
Query OK, 50 rows affected (0.02 sec)
Records: 50  Duplicates: 0  Warnings: 0
```



```
mysql> SELECT * FROM BSRFactTable;
+----------+-------------+-------------+----------+-------------+
| time_key | country_key | product_key | qty_sold | sales_value |
+----------+-------------+-------------+----------+-------------+
|      190 |         840 |           4 |        8 |          40 |
|      191 |         124 |          18 |       10 |      161.75 |
|      192 |         124 |           1 |        8 |       32.41 |
|      193 |         826 |          10 |        7 |       95.32 |
|      194 |         124 |           6 |        8 |       62.38 |
|      195 |         124 |           2 |        7 |       27.99 |
|      196 |         124 |           9 |       10 |       79.83 |
|      197 |         840 |           6 |        9 |          90 |
|      198 |         826 |          10 |        7 |       96.82 |
|      199 |         826 |           8 |        7 |        99.1 |
|      200 |         840 |           3 |        9 |          45 |
|      201 |         124 |          10 |        8 |       64.72 |
|      202 |          36 |           2 |        6 |       22.14 |
|      203 |         124 |           3 |        8 |       31.86 |
|      204 |          36 |           7 |        7 |       51.39 |
|      205 |         840 |          15 |        7 |         105 |
|      206 |         124 |          18 |       10 |      160.78 |
|      207 |          36 |           1 |        6 |       22.34 |
|      194 |         124 |          15 |        7 |       81.87 |
|      208 |          36 |           5 |        8 |       29.18 |
|      209 |         124 |          18 |        6 |       98.72 |
|      210 |         124 |          10 |        6 |       49.65 |
|      211 |         840 |           1 |        9 |          45 |
|      212 |         840 |           4 |        6 |          30 |
|      213 |         826 |          18 |        6 |      164.09 |
|      214 |          36 |           5 |        7 |        26.1 |
|      215 |         124 |           3 |        7 |       28.42 |
|      216 |         826 |           5 |        8 |       55.03 |
|      217 |         840 |           6 |        6 |          60 |
|      218 |          36 |           3 |        8 |       29.46 |
|      219 |         826 |          15 |        5 |      102.12 |
|      220 |          36 |          18 |        7 |      108.36 |
|      221 |          36 |           2 |        9 |       33.19 |
|      222 |          36 |           9 |        8 |       61.69 |
|      223 |         124 |           6 |        8 |       65.81 |
|      219 |         840 |          13 |        7 |         105 |
|      224 |          36 |           5 |        5 |       18.78 |
|      225 |         124 |           5 |        9 |       35.09 |
|      226 |         840 |           9 |        6 |          60 |
|      227 |          36 |          10 |        9 |       65.25 |
|      228 |          36 |           7 |        7 |        54.1 |
|      229 |         124 |          17 |        6 |       95.96 |
|      230 |          36 |          13 |        5 |        53.6 |
|      231 |         826 |           2 |        8 |       54.82 |
|      212 |          36 |           9 |       10 |       73.75 |
|      232 |         840 |          17 |        5 |         100 |
|      233 |         826 |           8 |        8 |      111.07 |
|      198 |         124 |           9 |        7 |       56.84 |
|      234 |         840 |           2 |        7 |          35 |
|      202 |         826 |           3 |        6 |       41.66 |
+----------+-------------+-------------+----------+-------------+
50 rows in set (0.00 sec)
```

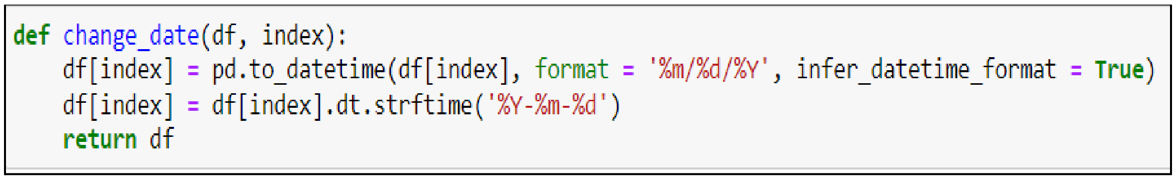
Actions Taken

Working on the Google Cloud Platform, our team (three members) was able to create a MySQL instance, along with two user access' in addition to the root user. In this way we were each able to connect to the cloud shell and collaborate to build out the databases and tables required simultaneously. The screenshots above show the specific MySQL functions and queries used to complete the ETL process between the two databases.

After creating the ODS database and all of the tables outlined in the schema provided, we imported the provided Countries.csv, CountryCurrencies.csv, and Products.csv. For the *Products* table we also added additional products to the table to have more variety within the dataset, for a total of 18 products. For all remaining ODS tables we created our own datasets before importing. We built a dataset that incorporates 20 records for *Customers* (with 20 corresponding CustomerLocations and *CustomerLocationAddresses*), three *AddressTypes*, 50 *SalesOrders,* 50 *SalesOrderAddresses,* 50 *SalesOrderLines* (one per order), and 50 entries for *ProductCountryPricing* that correspond to orders.

To perform the ETL process between the ODS database and the DW database we had a similar mixture of importing provided datasets, importing datasets that we built, and writing custom queries as well. The following tables were filled by independently creating and importing a corresponding dataset which had a numeric key corresponding to a date aspect specified by the table title: *YearDimension,WeekOfYearDimension, MonthOfYearDimension, DayOfMonthDimension, DayOfWeekDimension.* The *CountryDimension* and *ProductDimension* tables were populated with columns from the *Countries* and *Products* tables in the ODS database, as shown in screenshots above. The *CurrencyConversion* table was populated with a dataset provided to us, and then additional steps were taken to import currency exchange rates from Australian Dollar to US Dollar as well since we had created customer locations in Australia. The queries for filling and calculating the *TimeDimension* and *BSRFactTable* tables are documented in screenshots above and contain several references to many tables previously mentioned.

Importantly, within both databases all tables that contain a date field had to undergo a cleaning process before the appropriate dataset could be imported. In Python, our team set up a script to read in a csv file, convert specified columns to 'datetime' format that matches Excel date format of '%m/%d/%Y', and convert the column to from this datetime format, to another datetime format which matches the MySQL date format.



```python
def change_date(df, index):
    df[index] = pd.to_datetime(df[index], format = '%m/%d/%Y', infer_datetime_format = True)
    df[index] = df[index].dt.strftime('%Y-%m-%d')
    return df
```

Without taking this action, dates would fail to import and be read in 0000-00-00 which was a clear problem with foreign key relationships and ensuring tables were properly referencing each other.

Productionalizing The Process
Defining, accessing, wrangling, analyzing, visualizing, and organizing the data are all steps that need to be performed to achieve the process of production. As new data is created in the ODS database, there are a few steps to complete in order to retain historical reports. Checking the last time an event or batch was run is a good way to see how long ago the database was accessed. Grabbing dates after the last date from the first batch will ensure that previous data will not be overwritten. Creating a date_added column, as we have for this assignment, allows for the dates to be organized. This also permits data to be stored in an organized manner letting future team members sort through the data with ease. At the current moment, our databases are marked by calendar date alone. There is not a timestamp designating what time of day orders occurred. For future use of the database, either a) a timestamp needs to be added or b) a policy needs to be established to only import new orders spanning from a selected prior date to “yesterday”’s date. This ensures that either a) a more precise cutoff point of which orders to import can be established or b) we guarantee that we are only importing a fully completed day’s worth of orders. While sales orders and customers are expected to be the most common additions to the database, new products may be added or even currencies and countries can be created on rare occasions. It is important to ensure all tables are being updated and joined properly so that child keys are not faced with an error due to out of date parent keys.

A tangible step that we took in this project to aid future productionalization was to populate the BSRFactTable without the need for a temporary table through the use of CASE function in our query. In the future with any large datasets needing to be updated this will significantly speed up the processing time. This can be triggered to auto update weekly/monthly/etc based on whatever frequency the company BSR would like to set as their standard.

In our country and product dimensional tables in the DW database we included all countries and all products, not only countries and products that were included in sales orders at this time. This will eliminate the need for updating these tables each time there is a sales order to a new location or for a first-time new product. All that will be necessary in the future is to import currency exchange rates and unit pricing for the currency and product in question. This information can be added to the ODS database and have ETL queries run as needed to bring this information over into the DW database.

Difficulties Faced
The largest difficulty faced was properly formatting and cleaning our source data. The dates added, inactivated, ordered, cancelled, or completed in various tables all required proper formatting before they could be imported in the database. This required action taken outside of the database and MySQL commands. In Python we were able to resolve this issue as explained previously.

Another challenge was properly calculating sales value of orders placed on weekends throughout the year. As there are no directly corresponding currency exchange rates for Saturdays and Sundays we needed a way to properly determine this value. As seen in screenshots, we made use of the day_of_week_key in the *TimeDimension* table to take action only on weekend sales orders and associate these orders with a *CurrencyConversion* valuation_date of the preceding Friday rather than the actual order date. For this to work across all orders we had to import exchange rates for Australian Dollar (AUD) to US Dollar (USD), as this was not provided in our dataset, for customers located in Australia. This issue was quickly resolved but is an important reminder for companies to prepare to have data on file before they need it as we initially faced NULL sales_value results for orders from Australia.

A final, minor, difficulty was creating tables in an order that makes the most logical sense. We ended up creating a small amount of extra work by creating tables containing foreign keys before creating tables that contained the primary key of those primary keys, and were faced with errors by referencing non-existent tables. This could be avoided by inspecting the schema and determining the most efficient path forward.