**GROUP 5**

**Sales Representative DB**

Nelson Liang

Victor Li

Udit Manocha

Steven Gonzalez

**Overview**

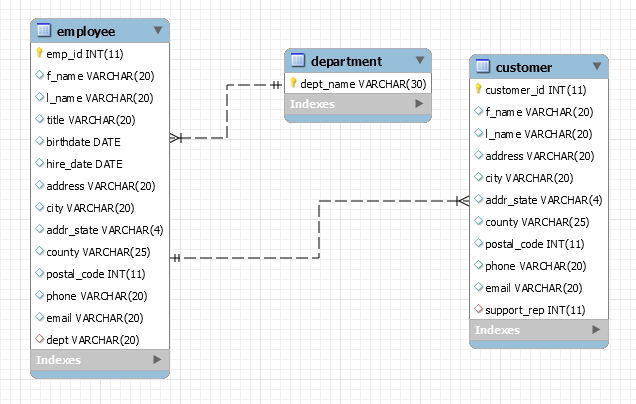
The purpose of this project is to implement a Relational Database Management System (RDBMS) that will Demonstrate Insertions, Updates and Deletions while expressing constraints to the database. The database will be populated with enough information in order to allow Querying of the database along with the proposed UIDs.

**Project Idea**

Our group has decided to develop a database based on three tables: Employees, Departments, and Customers. The Employee table will consist of 13 attributes of various data types. The attributes are as follows: EmployeeID (Primary Key), FirstName, LastName, JobTitle, BirthDate, HireDate, Address, City, State, Country, ZipCode, Phone, Email, and Department. The second table, Departments, will consist of the four following attributes: DepartName (Primary Key), DepartHead, and the Location. Lastly, the Customers table will consist of the following 11 attributes: CustomerID (Primary Key), FirstName, LastName, Address, City, State, Country, ZipCode, Phone, Email, and SupportRepID.

As a group we have determined that the number of tuples in the Customer table should exceed that of the Employee table. The number of tuples in the Employees table will also exceed that of the Departments table. As such, there can be multiple customers assign to a single Employee, however, no customer may have more than one employee. In addition, each department can have multiple employees, however, employees may not have multiple departments.

**E-R Diagram**



**View**

**Create a View**

/\* Find all customers in customer table that are from Utah and create a view with their customer ID and last names \*/

*CREATE OR REPLACE VIEW v AS*

*SELECT \* FROM customer*

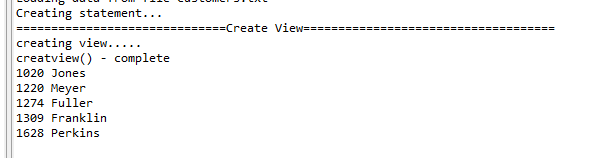
*WHERE customer.addr\_state = 'Utah';*

**Simple query on the View**

/\* select everything from view v \*/

SELECT \* from v

//OUTPUT: SELECT \* from v



**Inserts**

**INSERT - 1**

INSERT INTO employee

(f\_name, l\_name, emp\_id, title, address, city, addr\_state, country, postal\_code, phone, email, dept) VALUES ('Steven', 'Gonzalez', 120, 'Software Engineer', '1 Washington Square Hall', 'San Jose', 'California', 'United States', 95023, '1-(831)801-8637', 's.gonzalez0250@yahoo.com', 'Software');

**INSERT - 2**

INSERT INTO project2\_db.customer

(`customer\_id`, `f\_name`, `l\_name`, `address`, `city`, `addr\_state`, `country`, `postal\_code`, `phone`, `email`) VALUES ('1989', 'Peter ', 'Pan', '52127 Del Mar Crossing', 'Arvada', 'Colorado', 'United States', '19160', '1-(201)927-1717', 'wrobertsd123@bluehost.com');

**INSERT - 3**

INSERT into Department

(dept\_name, dept\_address, dept\_city, dept\_state, dept\_country, dept\_lead) VALUES ('electronics', 'Miller Rd', 'Hollister', 'California', 'US', 'Jim Jones');

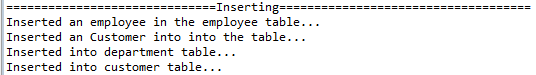
**INSERT - 4**

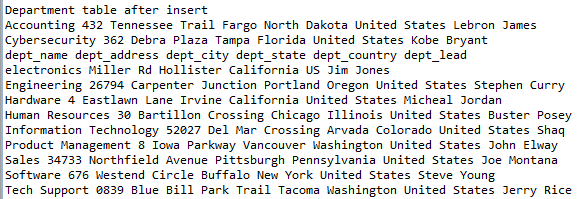
INSERT INTO customer

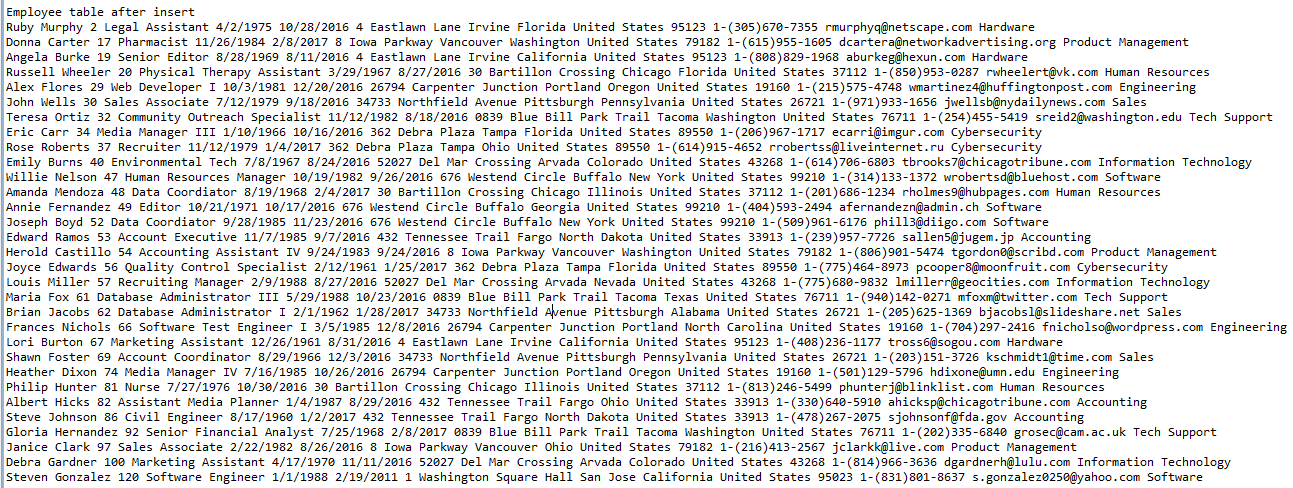
(customer\_id,f\_name,l\_name,address,city,addr\_state,country,postal\_code,phone,email,support\_rep)

VALUES (2000, 'Udit', 'Manocha', 'One Washington Sq', 'San Jose','California', 'United States', 95112, '1-(424)288-0824', ['udit.manocha@sjsu.edu](mailto:'udit.manocha@sjsu.edu)', 56);

//OUTPUT: after execution of insert statements







//customer table after insert execution



**Updates**

**UPDATE-1**

UPDATE project2\_db.customer

SET `address` = '111 Hoffman Trail'

WHERE `customer\_id` = '1989';

**UPDATE-2**

UPDATE employee

SET l\_name = 'Barrientos'

WHERE f\_name = 'Rose';

**UPDATE-3**

UPDATE department

SET dept\_lead = 'Babe Ruth'

WHERE dept\_name = 'electronics';

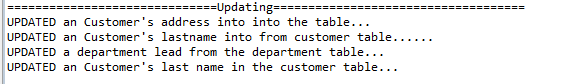
**UPDATE-4**

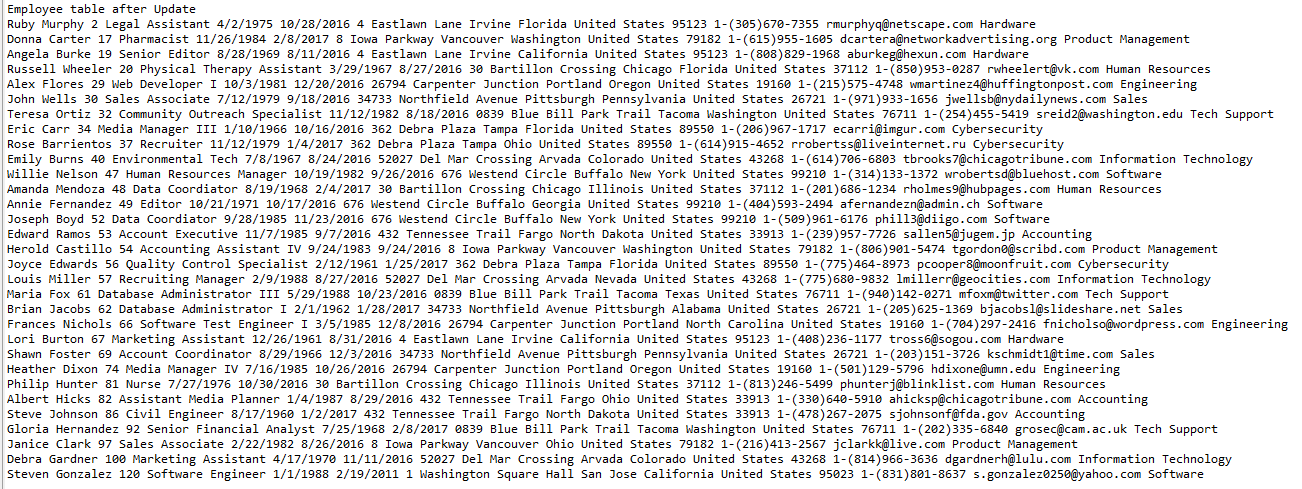
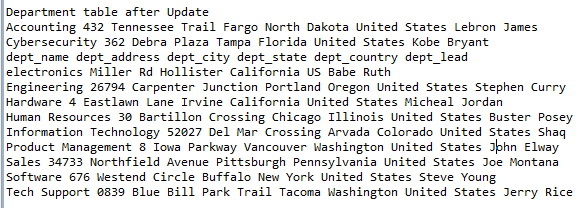
UPDATE customer

SET l\_name = 'Sharma'

WHERE customer\_id = 2000;

//OUTPUT: after execution of insert









**Deletions**

**DELETE-1**

DELETE FROM employee

WHERE emp\_id = 2;

**DELETE-2**

DELETE FROM project2\_db.customer

WHERE `customer\_id`='1989';

**DELETE-3**

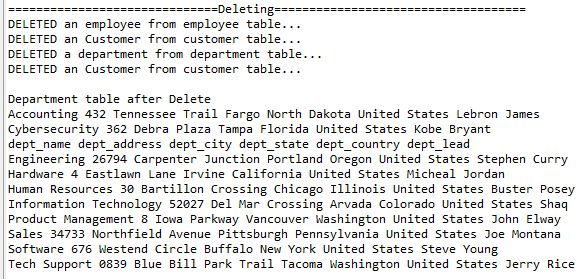
DELETE FROM department

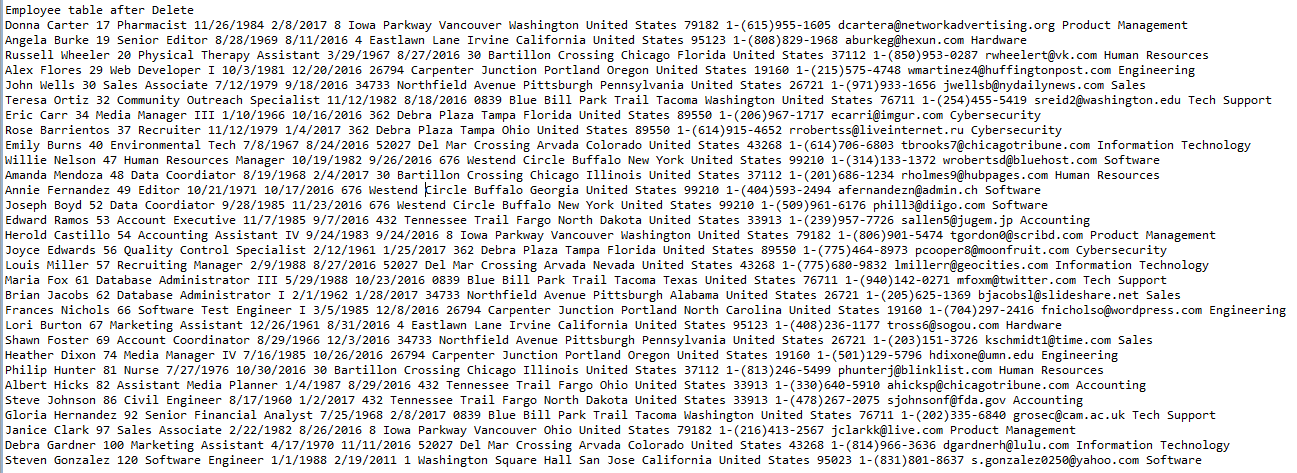
WHERE `dept\_name` = 'electronics';

**DELETE-4**

DELETE FROM customer

WHERE customer\_id = 1989

//OUTPUT: After execution of delete 







**Aggregation and Nested Queries**

**COUNT – the number of customers each employee has**

SELECT employee.f\_name, employee.l\_name, support\_rep AS rep\_id, COUNT(support\_rep)

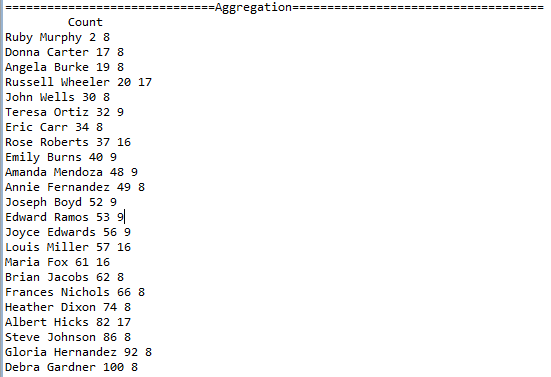
FROM customer JOIN employee

ON customer.support\_rep = employee.emp\_id

GROUP BY support\_rep

ORDER BY support\_rep ASC;

OUTPUT: count the number of customers each employee has



**MIN – find out who has least number of customers get rep\_id and their number of customers**

SELECT rep\_id, MIN(myCount)

FROM (SELECT support\_rep AS rep\_id, COUNT(support\_rep) AS myCount

FROM customer

GROUP BY support\_rep) AS countFunction;

//OUTPUT: sales rep id who has the minimum customer accounts



**MAX – find out who has most customers, get rep\_id and number of customers**

SELECT rep\_id, MAX(myCount)

FROM (SELECT support\_rep AS rep\_id, COUNT(support\_rep) AS myCount

FROM customer

GROUP BY support\_rep) AS countFunction;

//OUTPUT: sales rep id who has the most customer accounts



**AVG – find out the average customers each employee has**

SELECT rep\_id, AVG(myCount)

FROM (SELECT support\_rep AS rep\_id, COUNT(support\_rep) AS myCount

FROM customer

GROUP BY support\_rep) AS countFunction;

//OUTPUT: average number of customers each employee has



**SUM – count sales rep for each customer to determine how many customers in customer table**

SELECT SUM(myCount)

FROM (SELECT support\_rep AS rep\_id, COUNT(support\_rep) AS myCount

FROM customer

GROUP BY support\_rep) AS countFunction;

//OUTPUT: number of customers in the customer table

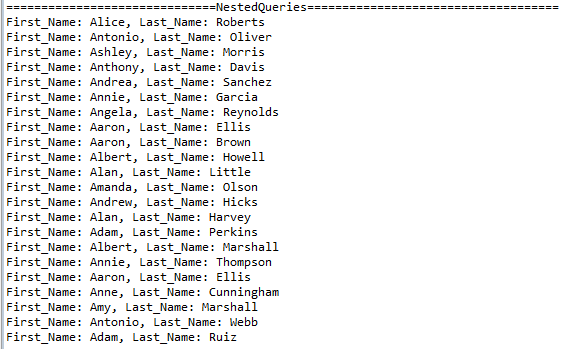


**/\* Select first and last names of all customers whos first name starts with the letter ‘A’ \*/**

SELECT f\_name, l\_name from customer

WHERE f\_name IN (SELECT f\_name FROM customer WHERE f\_name LIKE 'A%');

OUTPUT: first and last names of all customers whose first name starts with ‘A’

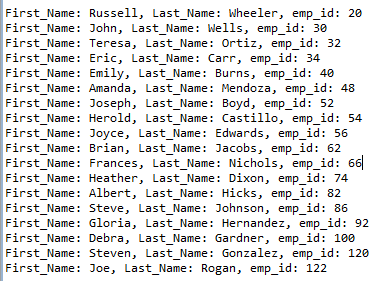


**/\* Select first name, last name and employee id of all employees in employee table with even numbered employee ids \*/**

SELECT f\_name, l\_name, emp\_id FROM employee

WHERE emp\_id IN (SELECT emp\_id FROM employee WHERE emp\_id MOD 2 = 0);

//OUTPUT: first and last names along with employee ids of employees who id number is even



**/\* get the department names, address, city, state and department lead of all departments located in California \*/**

SELECT dept\_name, dept\_address, dept\_city, dept\_state, dept\_lead

FROM department WHERE dept\_state IN (SELECT dept\_state FROM department

WHERE dept\_state = 'California');

//OUTPUT: department name, address, city, state and dept\_lead of deparments located in ‘California’



**Two and Three-Way Join**

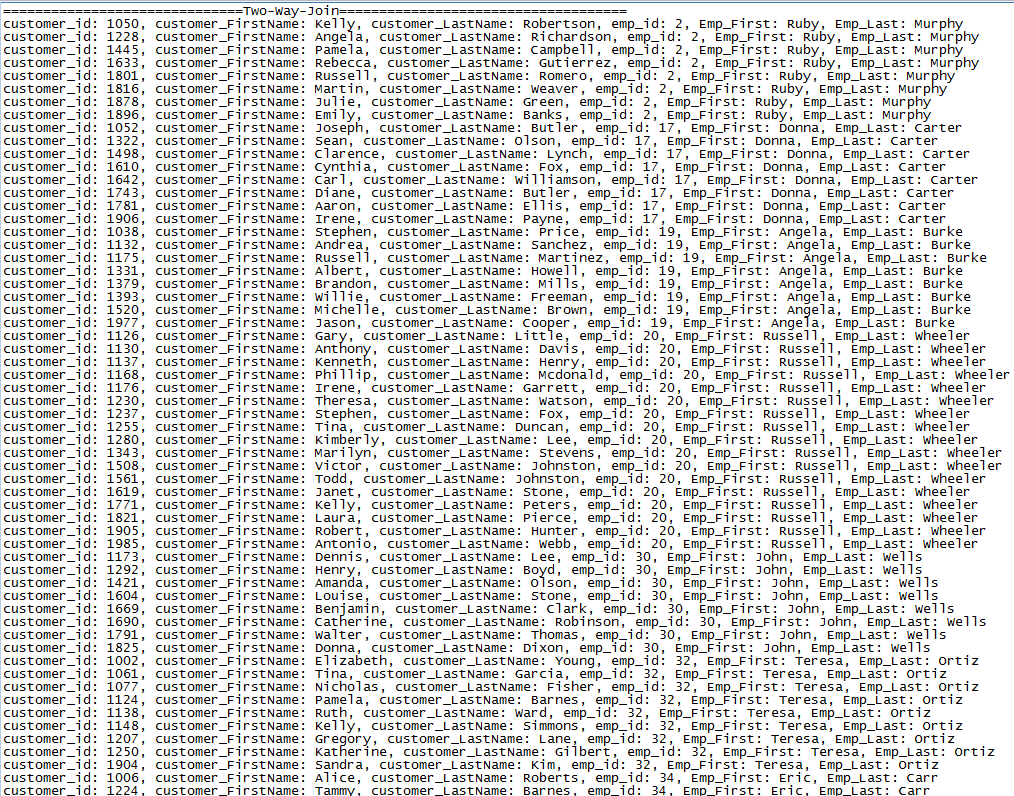
**Two-Way Join**

SELECT customer.customer\_id, customer.f\_name, customer.l\_name, employee.emp\_id, employee.f\_name, employee.l\_name

FROM project2\_db.`customer` INNER JOIN project2\_db.`employee`

ON customer.support\_rep = employee.emp\_id;

//OUTPUT: Two Way Join – NOTE 🡪 table was so big all of it wasn’t included



**Three-Way Join**

SELECT customer.f\_name, customer.l\_name, employee.f\_name, employee.l\_name, department.dept\_name, department.dept\_lead

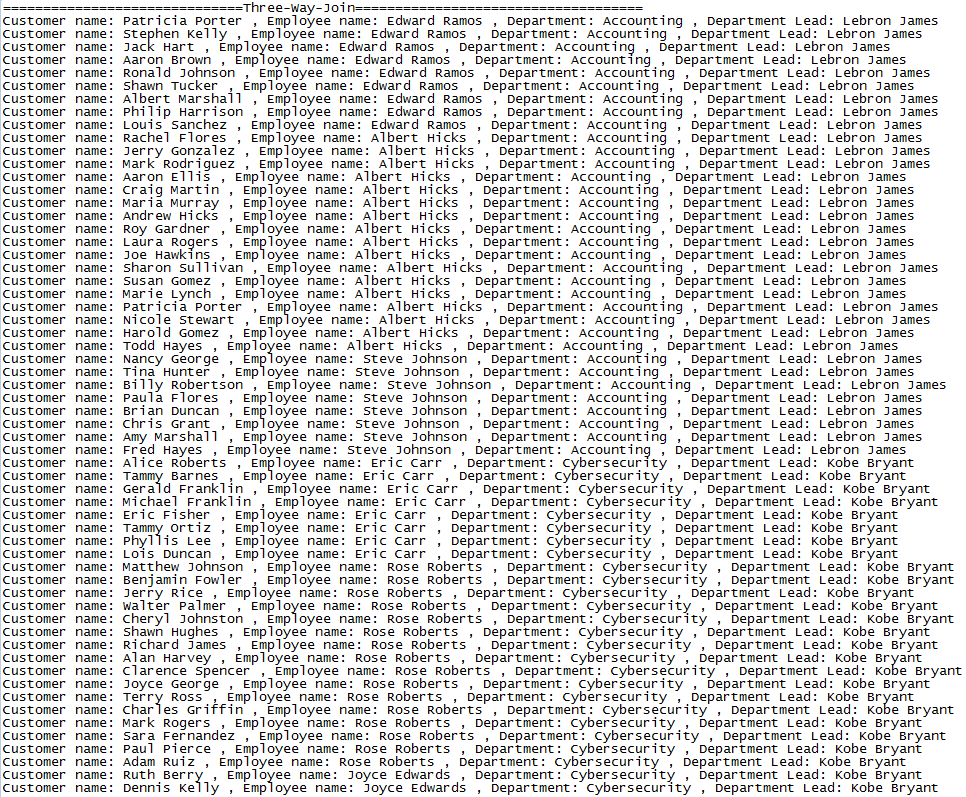
FROM employee

JOIN customer ON employee.emp\_id = customer.support\_rep

JOIN department ON department.dept\_name = employee.dept

ORDER BY dept\_name;

//OUTPUT: Three Way Join -- NOTE 🡪 table was so big all of it wasn’t included



**Transaction**

**Transaction with Constraint Violation**

**public** **static** **void** transactionWithViolation() **throws** SQLException {

**try** {

*conn*.setAutoCommit(**false**); //turn off auto-commit

*statement* = *conn*.createStatement();

*sql* = "INSERT INTO employee (f\_name, l\_name, emp\_id, title, address, city, addr\_state, country, postal\_code, phone, email, dept) " + "VALUES ('Steven', NULL, 120, NULL, '1 Washington Square Hall', 'San Jose', 'California', 'United States', 95023, '1-(831)801-8637', 's.gonzalez0250@yahoo.com', 'Software');";

*statement*.executeUpdate(*sql*);

//Never reached because first insert statement trips a constraint violation and jumpt to catch block

*sql* = "INSERT INTO employee (f\_name, l\_name, emp\_id, title, address, city, addr\_state, country, postal\_code, phone, email, dept) " + "VALUES ('Steven', NULL, 122, NULL, '1 Washington Square Hall', 'San Jose', 'California', 'United States', 95023, '1-(831)801-8637', 's.gonzalez0250@yahoo.com', 'Software');";

*statement*.executeUpdate(*sql*);

*conn*.commit(); // never reached

}**catch**(SQLException e) {

//constraint violation... so rollback changes

System.***out***.println("Constraint violation - Columns 'l\_name' and 'title' in employee table cannot be NULL. processing Rollback");

System.***out***.println("Processing Rollback");

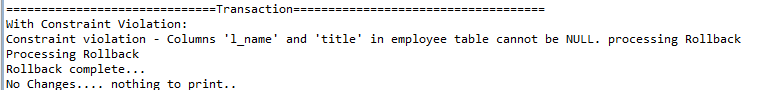
*conn*.rollback();

System.***out***.println("Rollback complete...");

}

}

//OUTPUT: Transaction with constraint violation



**Transaction No Violation**

**public** **static** **void** transactionNoViolation() **throws** SQLException {

**try** {

*conn*.setAutoCommit(**false**);

*statement* = *conn*.createStatement();

//no violation

*sql* = "INSERT INTO employee (f\_name, l\_name, emp\_id, title, address, city, addr\_state, country, postal\_code, phone, email, dept) " + "VALUES ('Joe', 'Rogan', 122, 'Engineer', '1 Washington Square Hall', 'San Jose', 'California', 'United States', 95023, '1-(831)801-9999', 'joe111110@yahoo.com', 'Software');";

*statement*.executeUpdate(*sql*);

*conn*.commit(); //changes are committed

System.***out***.println("Successful Transaction commit");

}**catch**(SQLException e) {

//Never reached, no violation in try block

System.***out***.println("Constraint violation - processing Rollback");

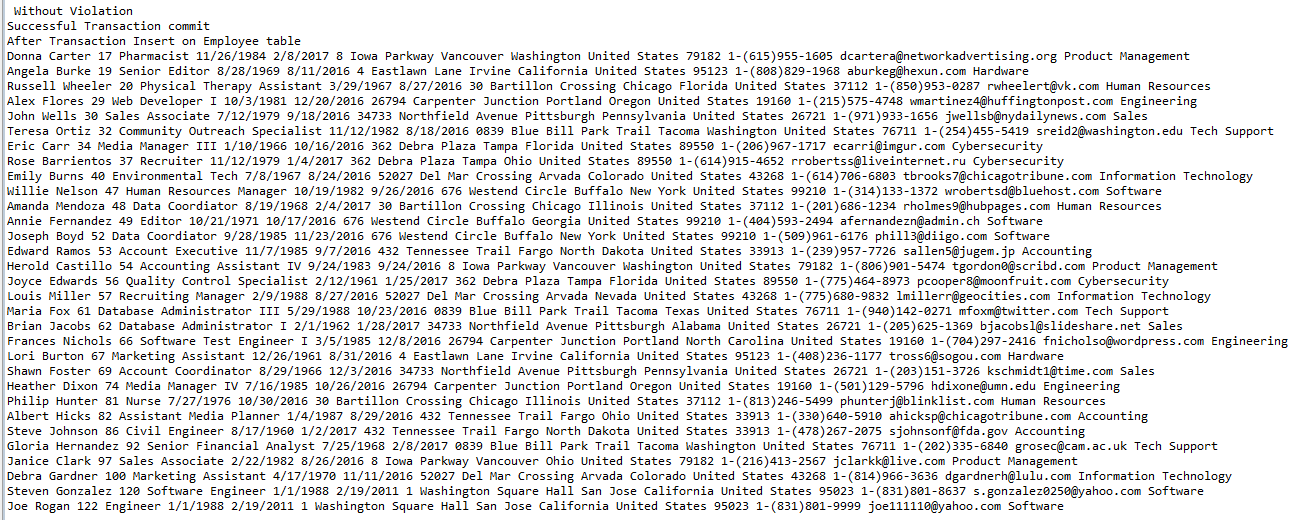
*conn*.rollback();

System.***out***.println("Rollback complete...");

}

}

//OUTPUT: Transaction without any violation, successful commit, last entry of the table “Joe Rogan was the tuple added in the transaction”



**JDBC – CODE**

**import** java.sql.\*;

**public** **class** project2\_db {

// JDBC driver name and database URL

**static** **final** String ***JDBC\_DRIVER*** = "com.mysql.jdbc.Driver";

**static** **final** String ***DB\_URL*** = "jdbc:mysql://localhost/";

// Database credentials

**static** **final** String ***USER*** = "root";

**static** **final** String ***PASS*** = "cs157a"; // insert your MySQL password

**private** **static** Connection *conn* = **null**;

**private** **static** PreparedStatement *preparedStatement* = **null**;

**private** **static** Statement *statement* = **null**;

**private** **static** ResultSet *rs* = **null**;

**private** **static** String *sql* = **null**;

**private** **static** project2\_db *db*;

**public** **static** **void** main(String[] args) **throws** SQLException {

*db* = **new** project2\_db();

}

**public** project2\_db() **throws** SQLException {

**try** {

Class.*forName*(***JDBC\_DRIVER***); //Register JDBC Driver

//STEP 3: Open a connection

System.***out***.println("Connecting to a selected database...");

*conn* = DriverManager.*getConnection*(***DB\_URL***, ***USER***, ***PASS***);

System.***out***.println("Connected database successfully...");

*createDatabase*();

*createTables*();

*loadIntoTables*();

//STEP 4: Execute a query

System.***out***.println("Creating statement...");

*statement* = *conn*.createStatement();

System.***out***.println("\n==============================Create View====================================");

ResultSet result = *createView*();

*printCustomer*(result);

//createTrigger();

System.***out***.println("\n==============================Two-Way-Join====================================");

*twowayjoin*();

System.***out***.println("\n\n==============================Three-Way-Join====================================");

*threeWayJoin*();

System.***out***.println("\n\n==============================Aggregation====================================");

System.***out***.println("\t Count");

*countAggregate*();

System.***out***.println("\n\t Min");

*minAggregate*();

System.***out***.println("\n\t Max");

*maxAggregate*();

System.***out***.println("\t AVG");

*avgAggregate*();

System.***out***.println("\t SUM");

*sumAggregate*();

System.***out***.println("\n==============================Inserting====================================");

*insert*();

//System.out.println("\n\tInsert2 method:");

//insertTwo();

System.***out***.println("\nDepartment table after insert");

*printDepartmentTable*();

System.***out***.println("\nEmployee table after insert");

*printEmployeeTable*();

System.***out***.println("\nCustomer table after insert");

*printCustomerTable*();

System.***out***.println("\n\n==============================Updating====================================");

*update*();

//System.out.println("\tUpdate2 method");

//updateTwo();

System.***out***.println("\nDepartment table after Update");

*printDepartmentTable*();

System.***out***.println("\nEmployee table after Update");

*printEmployeeTable*();

System.***out***.println("\ncustomer table after Update");

*printCustomerTable*();

System.***out***.println("\n\n==============================Deleting====================================");

*delete*();

//System.out.println("\tDelete2\n");

//deleteTwo();

System.***out***.println("\nDepartment table after Delete");

*printDepartmentTable*();

System.***out***.println("\nEmployee table after Delete");

*printEmployeeTable*();

System.***out***.println("\nCustomer table after Delete");

*printCustomerTable*();

System.***out***.println("\n\n==============================Transaction====================================");

System.***out***.println("With Constraint Violation: ");

*transactionWithViolation*();

System.***out***.println("No Changes.... nothing to print..\n\n Without Violation");

*transactionNoViolation*();

System.***out***.println("After Transaction Insert on Employee table");

*printEmployeeTable*();

System.***out***.println("\n");

System.***out***.println("\n\n==============================NestedQueries====================================");

*nestedQueries*();

*rs*.close();

}**catch**(SQLException se){

//Handle errors for JDBC

se.printStackTrace();

}**catch**(Exception e){

//Handle errors for Class.forName

e.printStackTrace();

}**finally**{

//finally block used to close resources

**try**{

**if**(*statement*!=**null**)

*conn*.close();

}**catch**(SQLException se){

}// do nothing

**try**{

**if**(*conn*!=**null**)

*conn*.close();

}**catch**(SQLException se){

se.printStackTrace();

}//end finally try

}//end try

System.***out***.println("\nGoodbye!");

}

**private** **static** **void** createDatabase() **throws** SQLException {

String queryDrop = "DROP DATABASE IF EXISTS project2\_db;";

Statement stmtDrop = *conn*.createStatement();

stmtDrop.execute(queryDrop);

*sql* = "SET foreign\_key\_checks = 0; ";

*statement* = *conn*.createStatement();

*statement* .execute(*sql*);

System.***out***.println("Creating database..."); // Create a database

String sql = "CREATE DATABASE project2\_db;";

*preparedStatement* = *conn*.prepareStatement(sql);

*preparedStatement*.executeUpdate();

System.***out***.println("Database created successfully...");

*conn* = DriverManager.*getConnection*(***DB\_URL***+"project2\_db", ***USER***, ***PASS***);

}

**private** **static** **void** createTables() **throws** SQLException {

String queryDrop = "DROP TABLE IF EXISTS department";

Statement stmtDrop = *conn*.createStatement();

stmtDrop.execute(queryDrop);

String createTableSQL = "CREATE TABLE department("

+ "dept\_name VARCHAR(40) NOT NULL PRIMARY KEY, "

+ "dept\_address VARCHAR(30), "

+ "dept\_city VARCHAR(40), "

+ "dept\_state VARCHAR(40), "

+ "dept\_country VARCHAR(40), "

+ "dept\_lead VARCHAR(40)) ";

*preparedStatement* = *conn*.prepareStatement(createTableSQL);

*preparedStatement*.executeUpdate();

System.***out***.println("Table called department created successfully...");

queryDrop = "DROP TABLE IF EXISTS employee";

stmtDrop = *conn*.createStatement();

stmtDrop.execute(queryDrop);

createTableSQL = "CREATE TABLE employee( "

+ "f\_name VARCHAR(20) NOT NULL, "

+ "l\_name VARCHAR(20) NOT NULL, "

+ "emp\_id INTEGER NOT NULL PRIMARY KEY, "

+ "title VARCHAR(30) NOT NULL, "

+ "birthdate VARCHAR(10) DEFAULT '1/1/1988', "

+ "hire\_date VARCHAR(10) DEFAULT '2/19/2011', "

+ "address VARCHAR(40), "

+ "city VARCHAR(20), "

+ "addr\_state VARCHAR(20), "

+ "country VARCHAR(25), "

+ "postal\_code INTEGER, "

+ "phone VARCHAR(20), "

+ "email VARCHAR(40), "

+ "dept VARCHAR(35), "

+ "FOREIGN KEY (dept) REFERENCES department(dept\_name)); ";

*preparedStatement* = *conn*.prepareStatement(createTableSQL);

*preparedStatement*.executeUpdate();

System.***out***.println("Table called employee created successfully...");

queryDrop = "DROP TABLE IF EXISTS customer";

stmtDrop = *conn*.createStatement();

stmtDrop.execute(queryDrop);

createTableSQL = "CREATE TABLE customer( "

+ "customer\_id INTEGER NOT NULL PRIMARY KEY, "

+ "f\_name VARCHAR(20), "

+ "l\_name VARCHAR(20), "

+ "address VARCHAR(40), "

+ "city VARCHAR(20), "

+ "addr\_state VARCHAR(20), "

+ "country VARCHAR(25), "

+ "postal\_code INTEGER, "

+ "phone VARCHAR(20), "

+ "email VARCHAR(40), "

+ "support\_rep INTEGER, "

+ "FOREIGN KEY (support\_rep) REFERENCES employee(emp\_id) "

+ "ON UPDATE CASCADE "

+ "ON DELETE CASCADE); ";

*preparedStatement* = *conn*.prepareStatement(createTableSQL);

*preparedStatement*.executeUpdate();

System.***out***.println("Table called customer created successfully...");

}

**public** **static** **void** loadIntoTables() **throws** SQLException {

String path = System.*getProperty*("user.dir").replace("\\", "\\\\") + "/departments.txt";

System.***out***.println("Loading data from file departments.txt");

String loadDataSQL = "LOAD DATA LOCAL INFILE '" + path + "' INTO TABLE department "

+ "LINES TERMINATED BY '\r\n'"; // need to add lines terminated if on windows

*preparedStatement* = *conn*.prepareStatement(loadDataSQL);

*preparedStatement*.execute();

path = System.*getProperty*("user.dir").replace("\\", "\\\\") + "/employees.txt";

System.***out***.println("Loading data from employees.txt");

loadDataSQL = "LOAD DATA LOCAL INFILE '" + path + "' INTO TABLE employee "

+ "LINES TERMINATED BY '\r\n'"; // need to add lines terminated if on windows

*preparedStatement* = *conn*.prepareStatement(loadDataSQL);

*preparedStatement*.execute();

path = System.*getProperty*("user.dir").replace("\\", "\\\\") + "/customers.txt";

System.***out***.println("Loading data from file customers.txt");

loadDataSQL = "LOAD DATA LOCAL INFILE '" + path + "' INTO TABLE customer "

+ "LINES TERMINATED BY '\r\n'"; // need to add lines terminated if on windows

*preparedStatement* = *conn*.prepareStatement(loadDataSQL);

*preparedStatement*.execute();

}

**public** **static** **void** countAggregate() **throws** SQLException {

*sql* = **null**;

*rs* = **null**;

*sql* = "SELECT employee.f\_name, employee.l\_name, support\_rep AS rep\_id, COUNT(support\_rep) "

+ "FROM customer JOIN employee "

+ "ON customer.support\_rep = employee.emp\_id "

+ "GROUP BY support\_rep "

+ "ORDER BY support\_rep ASC; ";

**try** {

*preparedStatement* = *conn*.prepareStatement(*sql*);

*rs* = *preparedStatement*.executeQuery();

**while**(*rs*.next()) {

String f\_name = *rs*.getString("f\_name");

String l\_name = *rs*.getString("l\_name");

**int** rep\_id = *rs*.getInt("rep\_id");

**int** count = *rs*.getInt("COUNT(support\_rep)");

System.***out***.println(f\_name + " " + l\_name + " " + rep\_id + " " + count);

}

}**catch**(SQLException e) { e.printStackTrace(); }

}

**public** **static** **void** minAggregate() **throws** SQLException {

*sql* = **null**;

*rs* = **null**;

*sql* = "SELECT rep\_id, MIN(myCount) "

+ "FROM (SELECT support\_rep AS rep\_id, COUNT(support\_rep) AS myCount "

+ "FROM customer "

+ "GROUP BY support\_rep) AS countFunction; ";

**try** {

*preparedStatement* = *conn*.prepareStatement(*sql*);

*rs* = *preparedStatement*.executeQuery();

**while**(*rs*.next()) {

**int** rep\_id = *rs*.getInt("rep\_id");

**int** minCount = *rs*.getInt("MIN(myCount)");

System.***out***.println("sales rep MIN(count): " + minCount);

}

}**catch**(SQLException e) { e.printStackTrace(); }

}

**public** **static** **void** maxAggregate() **throws** SQLException {

*sql* = **null**;

*rs* = **null**;

*sql* = "SELECT rep\_id, MAX(myCount) "

+ "FROM (SELECT support\_rep AS rep\_id, COUNT(support\_rep) AS myCount "

+ "FROM customer "

+ "GROUP BY support\_rep) AS countFunction; ";

**try** {

*preparedStatement* = *conn*.prepareStatement(*sql*);

*rs* = *preparedStatement*.executeQuery();

**while**(*rs*.next()) {

**int** rep\_id = *rs*.getInt("rep\_id");

**int** maxCount = *rs*.getInt("MAX(myCount)");

System.***out***.println("Sales Rep with the most Customer Contracts, Max(count): " + maxCount + "\n");

}

}**catch** (SQLException e) { e.printStackTrace(); }

}

**public** **static** **void** avgAggregate() **throws** SQLException {

*sql* = **null**;

*rs* = **null**;

*sql* = "SELECT rep\_id, AVG(myCount) "

+ "FROM (SELECT support\_rep AS rep\_id, COUNT(support\_rep) AS myCount "

+ "FROM customer "

+ "GROUP BY support\_rep) AS countFunction; ";

**try** {

*preparedStatement* = *conn*.prepareStatement(*sql*);

*rs* = *preparedStatement*.executeQuery();

**while**(*rs*.next()) {

**int** rep\_id = *rs*.getInt("rep\_id");

**int** avgCount = *rs*.getInt("AVG(myCount)");

System.***out***.println("sales\_reps have number of average customers , AVG(count): " + avgCount + "\n");

}

}**catch**(SQLException e) { e.printStackTrace(); }

}

**public** **static** **void** sumAggregate() **throws** SQLException {

*sql* = **null**;

*rs* = **null**;

*sql* = "SELECT rep\_id, SUM(myCount) "

+ "FROM (SELECT support\_rep AS rep\_id, COUNT(support\_rep) AS myCount "

+ "FROM customer "

+ "GROUP BY support\_rep) AS countFunction; ";

**try** {

*preparedStatement* = *conn*.prepareStatement(*sql*);

*rs* = *preparedStatement*.executeQuery();

**while**(*rs*.next()) {

**int** rep\_id = *rs*.getInt("rep\_id");

**int** sumCount = *rs*.getInt("SUM(myCount)");

System.***out***.println("total customer contracts , SUM(count): " + sumCount + "\n");

}

}**catch**(SQLException e) { e.printStackTrace(); }

}

**public** **static** **void** nestedQueries() {

*sql* = **null**;

**try** {

*sql* = "SELECT f\_name, l\_name from customer "

+ "WHERE f\_name IN (SELECT f\_name FROM customer WHERE f\_name LIKE 'A%'); ";

*preparedStatement* = *conn*.prepareStatement(*sql*);

*rs* = *preparedStatement*.executeQuery();

**while**(*rs*.next()) {

String f\_name = *rs*.getString("f\_name");

String l\_name = *rs*.getString("l\_name");

System.***out***.println("First\_Name: " + f\_name + ", Last\_Name: " + l\_name);

}

System.***out***.println("\n");

*sql* = "SELECT f\_name, l\_name, emp\_id FROM employee "

+ "WHERE emp\_id IN (SELECT emp\_id FROM employee WHERE emp\_id MOD 2 = 0);";

*preparedStatement* = *conn*.prepareStatement(*sql*);

*rs* = *preparedStatement*.executeQuery();

**while**(*rs*.next()) {

String f\_name = *rs*.getString("f\_name");

String l\_name = *rs*.getString("l\_name");

**int** emp\_id = *rs*.getInt("emp\_id");

System.***out***.println("First\_Name: " + f\_name + ", Last\_Name: " + l\_name + ", emp\_id: " + emp\_id);

}

System.***out***.println("\n");

*sql* = "SELECT dept\_name, dept\_address, dept\_city, dept\_state, dept\_lead "

+ "FROM department WHERE dept\_state IN (SELECT dept\_state FROM department WHERE dept\_state = 'California');";

*preparedStatement* = *conn*.prepareStatement(*sql*);

*rs* = *preparedStatement*.executeQuery();

**while**(*rs*.next()) {

String dept\_name = *rs*.getString("dept\_name");

String dept\_address = *rs*.getString("dept\_address");

String dept\_city = *rs*.getString("dept\_city");

String dept\_state = *rs*.getString("dept\_state");

String dept\_lead = *rs*.getString("dept\_lead");

System.***out***.println("dept\_name: " + dept\_name + ", address: " + dept\_address + ", city: " + dept\_city + ", state: " + dept\_state

+ ", dept\_lead: " + dept\_lead);

}

} **catch**(SQLException e) { e.printStackTrace(); }

}

**public** **static** **void** insert() **throws** SQLException {

*sql* = **null**;

**try**{

//insert 1

*sql* = "INSERT INTO employee (f\_name, l\_name, emp\_id, title, address, city, addr\_state, country, postal\_code, phone, email, dept) "

+ "VALUES ('Steven', 'Gonzalez', 120, 'Software Engineer', '1 Washington Square Hall', 'San Jose', 'California', 'United States', 95023, '1-(831)801-8637', 's.gonzalez0250@yahoo.com', 'Software'); ";

*statement*.executeUpdate(*sql*);

System.***out***.println("Inserted an employee in the employee table...");

//insert 2

*sql* = "INSERT INTO project2\_db.customer (`customer\_id`, `f\_name`, `l\_name`, `address`, `city`, `addr\_state`, `country`, `postal\_code`, `phone`, `email`) "

+ "VALUES ('1989', 'Peter ', 'Pan', '52127 Del Mar Crossing', 'Arvada', 'Colorado', 'United States', '19160', '1-(201)927-1717', 'wrobertsd123@bluehost.com');";

*statement*.executeUpdate(*sql*);

//STEP 2: Extract data from result set

System.***out***.println("Inserted an Customer into into the table...");

//insert 3

*sql* = "INSERT into Department (dept\_name, dept\_address, dept\_city, dept\_state, dept\_country, dept\_lead) VALUES ('electronics', 'Miller Rd', 'Hollister', 'California', 'US', 'Jim Jones');";

*statement*.executeUpdate(*sql*);

System.***out***.println("Inserted into department table... ");

//insert 4

*statement* = *conn*.createStatement();

*statement*.executeUpdate("INSERT INTO customer(customer\_id,f\_name,l\_name,address,city,addr\_state,country,postal\_code,phone,email,support\_rep)" + "VALUES (1302,'Udit','Manocha','One Washington Sq', 'San Jose','California','United States',95112,'1-(424)288-0824','udit.manocha@sjsu.edu',56)");

System.***out***.println("Inserted into customer table... ");

}**catch** (SQLException e) {e.printStackTrace();}

}

**public** **static** **void** update() **throws** SQLException {

*sql* = **null**;

**try** {

//update 1

*sql* = "UPDATE project2\_db.customer SET `address`='111 Hoffman Trail' WHERE `customer\_id`='1989';";

*statement*.executeUpdate(*sql*);

System.***out***.println("UPDATED an Customer's address into into the table...");

//update 2

*sql* = "UPDATE employee SET l\_name = 'Barrientos' where f\_name = 'Rose'; ";

*statement*.executeUpdate(*sql*);

System.***out***.println("UPDATED an Customer's lastname into from customer table......");

//update 3

*sql* = "UPDATE department SET dept\_lead = 'Babe Ruth' WHERE dept\_name = 'electronics' ";

*statement*.executeUpdate(*sql*);

System.***out***.println("UPDATED a department lead from the department table...");

//update 4

*statement* = *conn*.createStatement();

*statement*.executeUpdate("UPDATE customer SET l\_name = 'Sharma' WHERE customer\_id = 1989");

System.***out***.println("UPDATED an Customer's last name in the customer table...");

} **catch**(SQLException e) { e.printStackTrace(); }

}

**public** **static** **void** delete() **throws** SQLException {

*sql* = **null**;

**try** {

//delete 1

*sql* = "DELETE FROM employee WHERE emp\_id = 2; " ;

*statement*.executeUpdate(*sql*);

System.***out***.println("DELETED an employee from employee table...");

//delete 2

*sql* = "DELETE FROM project2\_db.customer WHERE `customer\_id`='1989';";

*statement*.executeUpdate(*sql*);

//STEP 2: Extract data from result set

System.***out***.println("DELETED an Customer from customer table...");

//delete 3

*sql* = "DELETE FROM department WHERE `dept\_name` = 'electronics';";

*statement*.executeUpdate(*sql*);

System.***out***.println("DELETED a department from department table...");

//delete 4

*statement* = *conn*.createStatement();

*statement*.executeUpdate("DELETE FROM customer WHERE customer\_id = 1971");

System.***out***.println("DELETED an Customer from customer table...");

} **catch**(SQLException e) { e.printStackTrace(); }

}

**public** **static** **void** twowayjoin() **throws** SQLException {

//STEP 1: Execute a query

**try**{

*sql* = "SELECT customer.customer\_id, customer.f\_name, customer.l\_name, employee.emp\_id, employee.f\_name, employee.l\_name "

+ "FROM project2\_db.`customer` INNER JOIN project2\_db.`employee` ON customer.support\_rep = employee.emp\_id;";

*rs* = *statement*.executeQuery(*sql*);

//STEP 2: Extract data from result set

**while**(*rs*.next()){

//Retrieve by column name

**int** cus\_id = *rs*.getInt("customer.customer\_id");

String cus\_first = *rs*.getString("customer.f\_name");

String cus\_last = *rs*.getString("customer.l\_name");

**int** emp\_id = *rs*.getInt("employee.emp\_id");

String emp\_first = *rs*.getString("employee.f\_name");

String emp\_last = *rs*.getString("employee.l\_name");

//Display values

System.***out***.print("customer\_id: " + cus\_id);

System.***out***.print(", customer\_FirstName: " + cus\_first);

System.***out***.print(", customer\_LastName: " + cus\_last);

System.***out***.print(", emp\_id: " + emp\_id);

System.***out***.print(", Emp\_First: " + emp\_first);

System.***out***.println(", Emp\_Last: " + emp\_last);

}

}**catch**(SQLException se) { se.printStackTrace(); }

}

**public** **static** **void** threeWayJoin() **throws** SQLException {

*sql* = **null**;

*rs* = **null**;

**try** {

*sql* = "SELECT customer.f\_name, customer.l\_name, employee.f\_name, employee.l\_name, department.dept\_name, department.dept\_lead "

+ "FROM employee "

+ "JOIN customer ON employee.emp\_id = customer.support\_rep "

+ "JOIN department ON department.dept\_name = employee.dept "

+ "ORDER BY dept\_name; ";

*rs* = *statement*.executeQuery(*sql*);

**while**(*rs*.next()){

String c\_f\_name = *rs*.getString("customer.f\_name");

String c\_l\_name = *rs*.getString("customer.l\_name");

String e\_f\_name = *rs*.getString("employee.f\_name");

String e\_l\_name = *rs*.getString("employee.l\_name");

String dept\_name = *rs*.getString("department.dept\_name");

String dept\_lead = *rs*.getString("department.dept\_lead");

System.***out***.println("Customer name: " + c\_f\_name + " " + c\_l\_name

+ " , Employee name: " + e\_f\_name + " " + e\_l\_name

+ " , Department: " + dept\_name + " , Department Lead: " + dept\_lead);

}

}**catch**(SQLException e) {

e.printStackTrace();

}

}

**public** **static** **void** transactionWithViolation() **throws** SQLException {

**try** {

*conn*.setAutoCommit(**false**); //turn off auto-commit

*statement* = *conn*.createStatement();

*sql* = "INSERT INTO employee (f\_name, l\_name, emp\_id, title, address, city, addr\_state, country, postal\_code, phone, email, dept) "

+ "VALUES ('Steven', NULL, 120, NULL, '1 Washington Square Hall', 'San Jose', 'California', 'United States', 95023, '1-(831)801-8637', 's.gonzalez0250@yahoo.com', 'Software');";

*statement*.executeUpdate(*sql*);

//Never reached because first insert statement trips a constraint violation and jumpt to catch block

*sql* = "INSERT INTO employee (f\_name, l\_name, emp\_id, title, address, city, addr\_state, country, postal\_code, phone, email, dept) "

+ "VALUES ('Steven', NULL, 122, NULL, '1 Washington Square Hall', 'San Jose', 'California', 'United States', 95023, '1-(831)801-8637', 's.gonzalez0250@yahoo.com', 'Software');";

*statement*.executeUpdate(*sql*);

*conn*.commit(); // never reached

}**catch**(SQLException e) {

//constraint violation... so rollback changes

System.***out***.println("Constraint violation - Columns 'l\_name' and 'title' in employee table cannot be NULL. processing Rollback");

System.***out***.println("Processing Rollback");

*conn*.rollback();

System.***out***.println("Rollback complete...");

}

}

**public** **static** **void** transactionNoViolation() **throws** SQLException {

**try** {

*conn*.setAutoCommit(**false**);

*statement* = *conn*.createStatement();

*sql* = "INSERT INTO employee (f\_name, l\_name, emp\_id, title, address, city, addr\_state, country, postal\_code, phone, email, dept) "

+ "VALUES ('Joe', 'Rogan', 122, 'Engineer', '1 Washington Square Hall', 'San Jose', 'California', 'United States', 95023, '1-(831)801-9999', 'joe111110@yahoo.com', 'Software');";

*statement*.executeUpdate(*sql*);

*conn*.commit();

System.***out***.println("Successful Transaction commit");

}**catch**(SQLException e) {

//Never reached, no violation in try block

System.***out***.println("Constraint violation - processing Rollback");

*conn*.rollback();

System.***out***.println("Rollback complete...");

}

}

**public** **static** **void** printCustomerTable() **throws** SQLException {

*rs* = **null**;

**try** {

*statement* = *conn*.createStatement();

*rs* = *statement*.executeQuery("SELECT \* from customer; ");

}**catch**(SQLException e) { e.printStackTrace(); }

**while**(*rs*.next()) {

**int** customer\_id = *rs*.getInt("customer\_id");

String f\_name = *rs*.getString("f\_name");

String l\_name = *rs*.getString("l\_name");

String address = *rs*.getString("address");

String city = *rs*.getString("city");

String state = *rs*.getString("addr\_state");

String country = *rs*.getString("country");

**int** zipcode = *rs*.getInt("postal\_code");

String phone = *rs*.getString("phone");

String email = *rs*.getString("email");

**int** support\_rep = *rs*.getInt("support\_rep");

System.***out***.println(customer\_id + " " + f\_name + " " + l\_name + " " + address + " "

+ city + " " + state + " " + country + " " + zipcode + " " + phone + " "

+ email + " " + support\_rep);

}

}

**public** **static** **void** printEmployeeTable() **throws** SQLException{

*rs* = **null**;

**try** {

*statement* = *conn*.createStatement();

*rs* = *statement*.executeQuery("SELECT \* from employee; ");

}**catch**(SQLException e) { e.printStackTrace(); }

**while**(*rs*.next()) {

String f\_name = *rs*.getString("f\_name");

String l\_name = *rs*.getString("l\_name");

**int** employee\_id = *rs*.getInt("emp\_id");

String title = *rs*.getString("title");

String birthdate = *rs*.getString("birthdate");

String hire\_date = *rs*.getString("hire\_date");

String address = *rs*.getString("address");

String city = *rs*.getString("city");

String state = *rs*.getString("addr\_state");

String country = *rs*.getString("country");

**int** zipcode = *rs*.getInt("postal\_code");

String phone = *rs*.getString("phone");

String email = *rs*.getString("email");

String department = *rs*.getString("dept");

System.***out***.println(f\_name + " " + l\_name + " " + employee\_id + " " + title + " "

+ birthdate + " " + hire\_date + " " + address + " " + city + " " + state + " "

+ country + " " + zipcode + " " + phone + " " + email + " " + department);

}

}

**public** **static** **void** printDepartmentTable() **throws** SQLException {

*rs* = **null**;

**try** {

*statement* = *conn*.createStatement();

*rs* = *statement*.executeQuery("SELECT \* from department; ");

}**catch**(SQLException e) { e.printStackTrace(); }

**while**(*rs*.next()) {

String department\_name = *rs*.getString("dept\_name");

String department\_address = *rs*.getString("dept\_address");

String department\_city = *rs*.getString("dept\_city");

String department\_state = *rs*.getString("dept\_state");

String department\_country = *rs*.getString("dept\_country");

String department\_lead = *rs*.getString("dept\_lead");

System.***out***.println(department\_name + " " + department\_address + " " + department\_city + " "

+ department\_state + " " + department\_country + " " + department\_lead);

}

}

**private** **static** **void** printCustomer(ResultSet rs) {

**try** {

**while**(rs.next()) {

System.***out***.print(rs.getString("customer\_id") + " " + rs.getString("l\_name") + "\n");

}

} **catch** (SQLException e) { e.printStackTrace(); }

}

**private** **static** ResultSet createView() {

System.***out***.println("creating view.....");

ResultSet rs = **null**;

**try** {

*statement* = *conn*.createStatement();

*statement*.executeUpdate("CREATE OR REPLACE VIEW v AS SELECT \* FROM customer WHERE customer.addr\_state = 'Utah';");

rs = *statement*.executeQuery("SELECT \* from v");

System.***out***.println("creatview() - complete");

}

**catch** (SQLException e) { e.printStackTrace(); }

**return** rs;

}

}