```
public class QueryBuilder {
	public static String collaboratorID;
	public static String researcherID;
	public static String labManagerID;
	public static String PIID;
	public static String labID;
	public static String projectName;
	public static boolean purchased = false;
	public static float totalFunding = 0;

	public static void setCollaboratorID(String cID) {
		collaboratorID = cID;
	}

	public static void setResearcherID(String rID) {
		researcherID = rID;
	}

	public static void setLabManagerID(String lmID) {
		labManagerID = lmID;
	}

	public static void setPIID(String piID) {
		PIID = piID;
	}

	public static void setLabID(String lID) {
		labID = lID;
	}

	public static void setProjectName(String pName) {
		projectName = pName;
	}

	public static void setTotalFunding(float funding) {
		totalFunding = funding;
	}

	public static void reset() {
		collaboratorID = null;
		researcherID = null;
		labManagerID = null;
		PIID = null;
		labID = null;
		projectName = null;
		purchased = false;
		totalFunding = 0;
	}

	public static void resetProject() {
		projectName = null;
		purchased = false;
		totalFunding = 0;
	}

	public static String getResearcherName() {
		return "Select name from Contains_LabMember cl, Researcher r " +
				"where r.id = cl.id and r.id = '" +
				researcherID + "'";
	}
```

```
	public static String getAllLabs() {
		return "Select * from Lab order by field";
	}

	public static String getAllLabMembers() {
		return "Select * from Contains_LabMember order by education";
	}

	public static String getAllResearchers() {
		return "Select * from Contains_LabMember cl, Researcher r where r.id = cl.id " +
				"order by researchertype";
	}

	public static String getAllLabManagers() {
		return "Select * from Contains_LabMember cl, LabManager lm where lm.id = cl.id " +
				"order by employmenttype";
	}

	public static String getAllPIs() {
		return "Select * from Contains_LabMember cl, PI pi where pi.id = cl.id " +
				"order by fieldofexpertise";
	}

	public static String getAllProjects() {
		return "select unique sw.projectName, pm.category, cl.name as PI " +
				"from Supervises_WorksOn sw, Contains_LabMember cl, Project_MaterialType
pm " +
				"where sw.piid = cl.id and pm.name = sw.projectName order by
pm.category";
	}

	public static String getPIName() {
		return "select cl.name from Supervises_WorksOn sw, Contains_LabMember cl " +
				"where sw.piid = cl.id and sw.projectName = '" + projectName + "'";
	}

	public static String getLabName() {
		return "select unique l.name from Supervises_WorksOn sw, Contains_LabMember cl, Lab l
" +
				"where sw.piid = cl.id and sw.projectName = '" + projectName + "' and
cl.labId = l.id";
	}

	public static String getLabNameByLabID() {
		return "select l.name from Lab l " +
				"where l.id = '" + labID + "'";
	}

	public static String getPINameByLabID() {
		return "select cl.name from Lab l, Contains_LabMember cl, PI p " +
				"where p.id = cl.id and cl.labId = l.id and l.id = '" + labID + "'";
	}

	public static String getAllResearchersForProject() {
		return "select * from researcher r, Supervises_WorksOn sw " +
				"where sw.rid = r.id and projectName = '" + projectName + "' " +
				"order by r.researcherType";
	}

	public static String countResearchersForProject() {
```

```
		return "select CAST(count(*) as INT) as count from researcher r, Supervises_WorksOn
sw " +
				"where sw.rid = r.id and projectName = '" + projectName + "'";
	}

	public static String getAllCollaborators() {
		return "select aci.name, aci.education " +
				"From Assigned_Collaborators_Id aci, Name_Education_ProjectName nep " +
				"Where aci.name = nep.name and nep.projectName = '" + projectName + "' "
+
				"order by aci.education";
	}

	public static String getMaterial() {
		return "select * from Project_MaterialType pm, MaterialType_MaterialPrice mm " +
				"where pm.materialType = mm.materialType and " +
				"pm.name = '" + projectName + "'";
	}

	public static String getAllBookings() {
		return "Select * from Takes_Booking where projectName = '" + projectName + "' " +
				"order by participanttestcondition";
	}

	public static String countControl() {
		return "Select cast(count(*) as int) as count from Takes_Booking where projectName =
'" +
				projectName + "' and " + "participanttestcondition LIKE '%Control%'";
	}

	public static String countExperimental() {
		return "Select cast(count(*) as int) as count from Takes_Booking where projectName =
'" +
				projectName + "' and " + "participanttestcondition LIKE '%Experimental
Group%'";
	}

	public static String calculateTotalFunding() {
		return "Select cast(SUM(amount) as float) as total " +
				"From Fund_ApprovedGrant f " +
				"Where f.projectName = '" + projectName + "'";
	}

	public static String calculateRemainingFunding() {
		return "Select projectName, (pf.total - pm.price) as remaining " +
				"from ProjectName_Funding pf, ProjectName_MaterialPrice pm " +
				"where pf.projectName = pm.name and pm.name = '" + projectName + "'";
	}

	public static String calculateWeeklyHours() {
		return "Select cast(sum(weeklyHoursAllocated) as int) as total " +
				"From Supervises_WorksOn sw, Role_WeeklyHoursAllocated rw " +
				"Where sw.rid = '" + researcherID + "' and rw.role = sw.role " +
				"Group by sw.rid";
	}

	public static String getAllGrants() {
		return "Select * " +
				"From Fund_ApprovedGrant f " +
				"Where f.projectName = '" + projectName + "'";
	}
```

```
	public static String countNumProject() {
		return "select cast(count(*) as int) as count from Project_MaterialType
p,Supervises_WorksOn sw, " +
				"Contains_LabMember cl where p.name = sw.projectName and sw.piid = cl.id
and "
				+ "cl.labId = '" + labID + "'";
	}

	public static String getProjectsGivenLab() {
		return "select p.name from Project_MaterialType p,Supervises_WorksOn sw, " +
				"Contains_LabMember cl where p.name = sw.projectName and sw.piid = cl.id
and "
				+ "cl.labId = '" + labID + "'";
	}

	public static String getPIName_simple() {
		return "Select cl.name from Contains_LabMember cl, PI p , Supervises_WorksOn sw " +
					"where p.id = cl.id and p.id = '" +
					PIID + "'";
	}

	public static String getLabManagerName() {
		return "Select name from Contains_LabMember cl, LabManager lm " +
				"where lm.id = cl.id and lm.id = '" +
				labManagerID + "'";
	}

	public static String getAllOpenGrants() {
		return "Select * from Applies_OpenGrant_Date " +
				"order by status";
	}

	public static String getApprovedOpenGrants() {
		return "Select * from Applies_OpenGrant_Date where status = 'approved'";
	}

	public static String getRejectedOpenGrants() {
		return "Select * from Applies_OpenGrant_Date where status = 'rejected'";
	}

	public static String getAppliedOpenGrants() {
		return "Select * from Applies_OpenGrant_Date where status = 'applied'";
	}

	public static String getGrantsForProject(String projectName_input) {
		return "select unique name, amount " +
				"from Fund_ApprovedGrant fa " +
				"where fa.projectName = '" +
				projectName_input + "'";
	}

	public static String getAllSubjects() {
		return "Select * from Subject order by id";
	}

	public static String getAllParticipants() {
		return "Select * from Participates order by sid";
	}

	public static String getAllProjectNames() {
```

```
		return "select unique sw.projectName " +
				"from Supervises_WorksOn sw, Contains_LabMember cl, Project_MaterialType
pm " +
				"where sw.piid = cl.id and pm.name = sw.projectName";
	}

	public static String getCountsByResearcherType() {
		return "select researcherType, count(*) " +
				"from Researcher r, Supervises_WorksOn w " +
				"where r.id = w.rid and w.projectName = '" + projectName + "' " +
				"group by researcherType";
	}

	public static String updateProjectCategory(String pname, String category) {
		return "update Project_MaterialType set category = '" + category.trim() + "' " +
"where name = '" + pname.trim() + "'";
	}

	public static String getAllCollaborators_simple() {
		return "select * from  Assigned_Collaborators_Id";
	}

	public static String onlyProjects() {
		return "select * from Project_MaterialType";
	}

	public static String getAvailableSubjects(String projectName_input, Date
dateParticipated_input, Integer startTime_input, Integer length_input) {
		return "select * from Subject s " +
		"where availability = 'Y' AND s.id NOT IN (" +
		getOverLapSubjects(projectName_input, dateParticipated_input, startTime_input)
		+ ")";

	}

	public static String getOverLapSubjects(String projectName_input, Date
dateParticipated_input, Integer startTime_input) {
		return "select p.sid from Participates p, Takes_Booking b " +
		"where b.projectName = '" + projectName_input + "' AND p.dateParticipated = '" +
		dateParticipated_input + "' AND (" + startTime_input + " <= p.startTime AND
p.startTime <= (" + startTime_input + "+ b.length))";
	}

	public static String getCollaboratorName() {
		return "Select name from Assigned_Collaborators_Id cl " +
				"where cl.id = '" +
				collaboratorID + "'";
	}

	public static String getCollaboratorProjects() {
		return "Select projectName " +
				"from Assigned_Collaborators_Id aci, Name_Education_ProjectName nep " +
				"where aci.name = nep.name and aci.id ='" +
				collaboratorID + "'";
	}
}
```

```
public class DBAPanelConstants {
	public static final String NAME = "Name";
	public static final String AMOUNT = "Amount";
	public static final String PROJ_NAME = "Project Name";
	public static final String STATUS = "Status";
	public static final String DATE_APP = "Date applied";
	public static final String ZERO = "0";
	public static final String LM_ID = "LM ID";
	public static final String LM_EDUC = "LM Education";
	public static final String LM_NAME = "LM Name";
	public static final String COLAB_ID = "COLAB ID";
	public static final String UPDATE_LAB_MEM_NAME = "lab_change_name";
	public static final String UPDATE_LAB_MEM_EDUC = "lab_change_educ";

/** Below are DBAdmin related tasks */

	public static String getAllOpenGrants() {
		return "select * from Applies_OpenGrant_Date";
	}

	public static String getAllApprovedGrants() {
		return "select * from Fund_ApprovedGrant";
	}

	public static String updateLabMemName()
	{
		return "update Contains_LabMember\r\n" +
				"set name = ?" +
				"where id = ?";
	}

	public static String updateLabMemEduc()
	{
		return "update Contains_LabMember set education = ? where id = ?";
	}

	public static String deleteCollaborator() {
		return "delete from Assigned_Collaborators_Id\r\n" +
				"where id = ?";
	}

	public static String deleteApprovedGrant() {
		return "delete from Applies_OpenGrant_Date    where status = 'approved'";
	}

	public static String approveSpecificGrant() {
		return "delete from Applies_OpenGrant_Date    where  = ?";
	}
}
```

```
public void insert_open_grant(String name, String amout_input, String date, String status) {
		System.out.println("insert_open_grant is called.");
		try {
```

```
                    PreparedStatement ps = null;
                    ps = con.prepareStatement("INSERT INTO Applies_OpenGrant_Date VALUES
(?,?,?, ?)");

                    ps.setString(1, name);
                    ps.setString(2, amout_input);
                    ps.setDate(3, new java.sql.Date(new java.util.Date().getTime()));
                    ps.setString(4, status);
                    int rowCount = ps.executeUpdate();
                    System.out.println(rowCount + " row got added.");
                    ps.close();
                    con.commit();

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

      public void insert_approved_grant(String name, String amount, String third_output) {
             System.out.println("insert_approved_grant is called.");
             try {
                    PreparedStatement ps = null;
                    ps = con.prepareStatement("INSERT INTO Fund_ApprovedGrant VALUES (?,?,?)");
                    ps.setString(1, name);
                    ps.setString(2, amount);
                    ps.setString(3, third_output);
                    int rowCount = ps.executeUpdate();
                    System.out.println(rowCount + " row got added.");
                    ps.close();

                    con.commit();

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

             public void update_open_grant_status(String name, String amout_input, String status)
{
                    System.out.println("insert_open_grant is called.");
                    try {
                           PreparedStatement ps = null;
                           ps = con.prepareStatement("UPDATE Applies_OpenGrant_Date set status = ?
where name = ? and amount = ?");

                           ps.setString(2, name);
                           ps.setString(3, amout_input);
                           ps.setString(1, status);
                           int rowCount = ps.executeUpdate();
                           System.out.println(rowCount + " row got added.");
                           ps.close();
                           con.commit();

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

```
private void insert(Connection con, String cid, String cname, String edu,String projectName) {
		int cid_int = Integer.parseInt(cid);
		PreparedStatement ps;
		PreparedStatement ps2;

		try{
			ps = con.prepareStatement("INSERT INTO Assigned_Collaborators_Id VALUES
(?,?,?)");
			ps2 = con.prepareStatement("INSERT INTO Name_Education_ProjectName VALUES
(?,?,?)");
			ps.setInt(1, cid_int);
		    ps.setString(2, cname);
		    ps.setString(3, edu);

		    ps2.setString(1,cname);
		    ps2.setString(2,edu);
		    ps2.setString(3,projectName);
		    ps.executeUpdate();
		    ps2.executeUpdate();

		    // commit work
		    con.commit();

		    ps.close();
		    ps2.close();
		} catch (Exception e) {
			System.out.println(e.getMessage());
		}
	}

private void insert2(Connection con, String projectName,String category,String materialType) {

		PreparedStatement ps;

		try{
			ps = con.prepareStatement("INSERT INTO Project_MaterialType VALUES (?,?,?)");
			ps.setString(1, projectName);
		    ps.setString(2, category);
		    ps.setString(3, materialType);

		    ps.executeUpdate();

		    // commit work
		    con.commit();

		    ps.close();
		} catch (Exception e) {
			System.out.println(e.getMessage());
		}
	}

private void insertSubjects(Connection con, String sid, String name, String availability) {
		PreparedStatement ps;

		try{
			ps = con.prepareStatement("insert into Subject VALUES (?,?,?)");
			ps.setString(1, sid);
		    ps.setString(2, name);
		    ps.setString(3, availability);

		    ps.executeUpdate();
```

```
                // commit work
                con.commit();

                ps.close();
            } catch (Exception e) {
                System.out.println(e.getMessage());
            }
        }

        private void insertAssignedCollaborators(Connection con, String cid, String name, String
education) {
            PreparedStatement ps;

            try{
                ps = con.prepareStatement("insert into Assigned_Collaborators_Id VALUES
(?,?,?)");
                ps.setString(1, cid);
                ps.setString(2, name);
                ps.setString(3, education);

                ps.executeUpdate();

                // commit work
                con.commit();

                ps.close();
            } catch (Exception e) {
                System.out.println(e.getMessage());
            }
        }

        private void insertNameEduProjName(Connection con, String name, String education, String
projectName) {
            PreparedStatement ps;

            try{
                ps = con.prepareStatement("insert into Name_Education_ProjectName VALUES
(?,?,?)");
                ps.setString(1, name);
                ps.setString(2, education);
                ps.setString(3, projectName);

                ps.executeUpdate();

                // commit work
                con.commit();

                ps.close();
            } catch (Exception e) {
                System.out.println(e.getMessage());
            }
        }

        private void insertTakesBooking(Connection con, String partNumber, String length, String
testCondition, String projectName) {
            PreparedStatement ps;
            Integer length_int = Integer.parseInt(length) * 100;
            Integer partNumber_int = Integer.parseInt(partNumber);

            try{
```

```
                ps = con.prepareStatement("insert into Takes_Booking VALUES (?,?,?,?)");
                ps.setString(1, projectName);
            ps.setInt(2, partNumber_int);
            ps.setInt(3, length_int);
                ps.setString(4, testCondition);

            ps.executeUpdate();

            // commit work
            con.commit();

            ps.close();
        } catch (Exception e) {
                System.out.println(e.getMessage());
        }
    }

    private void insertMakesBooking(Connection con, String partNumber, String projectName,
String lmid) {
        PreparedStatement ps;
        Integer partNumber_int = Integer.parseInt(partNumber);

        try{
                ps = con.prepareStatement("insert into Makes_Booking VALUES (?,?,?)");
                ps.setString(1, projectName);
            ps.setInt(2, partNumber_int);
            ps.setString(3, lmid);

            ps.executeUpdate();

            // commit work
            con.commit();

            ps.close();
        } catch (Exception e) {
                System.out.println(e.getMessage());
        }
    }

    private void insertParticipates(Connection con, String partNumber, String partDate, String
startTime, String sid, String projectName) {
        PreparedStatement ps;
        Integer partNumber_int = Integer.parseInt(partNumber);
        SimpleDateFormat formatter = new SimpleDateFormat("dd/MM/yyyy");
        try{
            Date partDate_dateJava = formatter.parse(partDate);
            Instant instant = partDate_dateJava.toInstant();
            ZoneId zoneId = ZoneId.of ( "America/Montreal" );
            ZonedDateTime zdt = ZonedDateTime.ofInstant (instant , zoneId);
            LocalDate partDate_localDate = zdt.toLocalDate();
            java.sql.Date partDate_dateSQL = java.sql.Date.valueOf(partDate_localDate);

                try{
                    ps = con.prepareStatement("insert into Participates VALUES
(?,?,?,?,?)");
                    ps.setString(1, sid);
                    ps.setString(2, projectName);
                    ps.setInt(3, partNumber_int);
                    ps.setDate(4, partDate_dateSQL);
                    ps.setString(5, startTime);
```

```
                    ps.executeUpdate();

                    // commit work
                    con.commit();

                    ps.close();
              } catch (Exception e) {
                    System.out.println(e.getMessage());
              }

        } catch (ParseException ex) {
              System.out.println("Message: " + ex.getMessage());
        }
}
```