Queries

Views

1. WHERE Clause

|  |
| --- |
| -- This view is used to display information technology jobs within a specific salary range  -- It provides a filtered list of IT jobs, including job title, description, salary, location, date posted, and closing date  -- The query below is used highly often for job searches, so this as a view allows the employment agency to give customers the information they need faster, as it uses fewer database resources  -- This view can also act as a template to help create views for other popular job sectors and salary ranges  CREATE VIEW information\_technology\_jobs AS  SELECT  job.job\_title AS 'Job Title',  job.job\_description AS 'Job Description',  job.salary AS 'Salary',  location.location\_name AS 'Location',  DATE(date\_posted) AS 'Date Posted',  job.closing\_date AS 'Closing Date'  FROM  job  INNER JOIN  location ON job.location\_id = location.location\_id  WHERE  sector\_id = 1  AND salary BETWEEN 60000 AND 80000; |

Querying the View:

|  |
| --- |
| SELECT  \*  FROM  information\_technology\_jobs |

Output:

A screenshot of a computer

Description automatically generated

1. Aggregate Functions

|  |
| --- |
| -- This view is used to show the count of job applications for each job  -- It presents information such as employer name, job title, sector, salary, and application count  -- This shows the employment agency which jobs are popular to apply for, helping them to decide which jobs to promote and advertise to get more new customers to use their services  -- Employers with jobs that have a low application count can be marketed a paid service to help promote their job posting to try increase applications and drive attention to it  CREATE VIEW job\_application\_count AS  SELECT  employer.employer\_name AS 'Employer',  job.job\_title AS 'Job',  sector.sector\_name AS 'Sector',  job.salary AS 'Salary',  COUNT(application.job\_id) AS 'Application Count'  FROM  job  LEFT JOIN  application ON job.job\_id = application.job\_id  INNER JOIN  sector ON job.sector\_id = sector.sector\_id  INNER JOIN  employer ON job.employer\_id = employer.employer\_id  GROUP BY job.job\_id  ORDER BY COUNT(\*) DESC , job.job\_id; |

Querying the View:

|  |
| --- |
| SELECT  \*  FROM  job\_application\_count |

Output:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

1. Joins

|  |
| --- |
| -- This view is used to display the status of candidates' applications for various jobs  -- It includes candidate name, job title, salary, sector, and application status  -- This gives the employment agency an overview of how successful their customers are at getting accepted for jobs  -- The employment agency can push helpful resources to unsuccessful customers to increase their chances of successfully getting a job, which means that if they ever need to apply for a new job in the future they will be more inclined to go through the business again as they had a previously positive experience  -- Candidates with a pending status can be marketed other job opportunities to keep them using the system  -- Successful candidates can be asked to review how they found the process through the employment agency  -- This can be filtered further to only show specific candidates  CREATE VIEW candidates\_application\_status AS  SELECT  candidate.first\_name AS 'First Name',  candidate.last\_name AS 'Last Name',  job.job\_title AS 'Job Title',  job.salary AS 'Salary',  sector.sector\_name AS 'Sector',  application.application\_status AS 'Application Status'  FROM  job  INNER JOIN  application ON job.job\_id = application.job\_id  INNER JOIN  candidate ON application.candidate\_id = candidate.candidate\_id  INNER JOIN  sector ON job.sector\_id = sector.sector\_id; |

Querying the View:

|  |
| --- |
| SELECT  \*  FROM  candidates\_application\_status |

Output:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

1. Filtering Records

|  |
| --- |
| -- This view is used to display jobs posted in the last 30 days  -- It provides a list of job titles, descriptions, sectors, salaries, date posted, and closing date  -- This gives the employment agency a list of all the newest job postings, which can be used in advertisements to help generate more traffic  -- As well, this view is a common and useful view to candidates looking for jobs so it can also be used by them, increasing the speed of the system and utilising fewer database resources    CREATE VIEW jobs\_posted\_in\_last\_30\_days AS  SELECT  job.job\_title AS 'Job Title',  job.job\_description AS 'Job Description',  sector.sector\_name AS 'Sector',  job.salary AS 'Salary',  DATE(job.date\_posted) AS 'Date Posted',  job.closing\_date AS 'Closing Date'  FROM  job  INNER JOIN sector ON job.sector\_id = sector.sector\_id  WHERE  date\_posted > NOW() - INTERVAL 30 DAY  AND job.closing\_date > CURDATE()  ORDER BY DATE(job.date\_posted) DESC; |

Querying the View:

|  |
| --- |
| SELECT  \*  FROM  jobs\_posted\_in\_last\_30\_days |

Output:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Java Database Connection

Making a Connection

|  |
| --- |
| // Class variable  private static Connection *con*;  // In main method  String host = "jdbc:mysql://jbothwell06.webhosting1.eeecs.qub.ac.uk";  String port = ":3306/";  String DB = "jbothwell06";  String username = "jbothwell06";  String password = "NVHMVw8mnqs9Yc7K";  String urlDB = host + port + DB;  *con* = DriverManager.*getConnection*(urlDB, username, password); |
|  |

Output:

A screenshot of a computer

Description automatically generated

Accessing a View

|  |
| --- |
| // Get the information\_technology\_jobs view  String statement = "SELECT \* FROM information\_technology\_jobs";  Statement stmt = *con*.createStatement();  ResultSet rs = stmt.executeQuery(statement);  // After creating table for display…  while (rs.next()) {  String jobTitle = rs.getString("Job Title");  String jobDescription = rs.getString("Job Description");  float salary = rs.getFloat("Salary");  String location = rs.getString("Location");  Date datePosted = rs.getDate("Date Posted");  Date closingDate = rs.getDate("Closing Date");  System.*out*.format(leftAlignFormat, jobTitle, jobDescription, salary, location, datePosted, closingDate);  } |
|  |

Output:

A screenshot of a computer

Description automatically generated

Inserting a New Record

|  |
| --- |
| // Carry out SQL INSERT statement with user input  Statement stmt = *con*.createStatement();  stmt.executeUpdate( "INSERT INTO candidate (first\_name, last\_name, address\_line, city, postal\_code, country, email, contact\_number, date\_of\_birth, currently\_seeking) VALUES ('" + firstName + "', '" + lastName + "', '" + addressLine + "', '" + city + "', '" + postalCode + "', '" + country + "', '" + email + "', '" + contactNumber + "', '" + dateOfBirth + "', '" + currentlySeeking + "')");    System.out.println( "\nRecord successfully inserted" );  // Close statement  stmt.close(); |
|  |

Output:

A screenshot of a computer

Description automatically generated