|  |  |
| --- | --- |
| **USE CASE 1** | Use the system to create a country report. |
| **Goal in Context** | A worker needs to create a report about Italy to use in their work. They need to display the code, name, continent, region, population and capital. |
| **Scope and Level** | Company, Summary. |
| **Preconditions** | We know the countries and have the system ready to use and connected to the MYSQL database. |
| **Success End Condition** | Worker has country report returned. |
| **Failed End Condition** | Worker was unable to return report. |
| **Primary, Secondary Actors** | Worker, database. |
| **Trigger** | Someone asking for the country report. |
| **Description** | 1. Company asks worker to get the report of a specific country. 2. Worker uses the system to return the country’s columns by entering its name into the return query. 3. Worker generates the report for the company to use. |

|  |  |
| --- | --- |
| **USE CASE 2** | Use the system to create a city report. |
| **Goal in Context** | A worker needs information about Edinburgh to use in their work. They need to display the name, the country it’s in, district and population. |
| **Scope & Level** | Company, Summary. |
| **Preconditions** | The list of cities is already available and known to us in the system. |
| **Success End Condition** | The worker receives a city report with the correct information. |
| **Failed End Condition** | The worker does not receive a city report and is left with no information. |
| **Primary, Secondary Actors** | Worker, Database. |
| **Trigger** | A need for a city report. |
| **Description** | 1. Worker is asked to get a report of a specific city and the information pertaining it. 2. Worker enters city name/ID into system. 3. Worker generates the report of the city from the system. |

|  |  |
| --- | --- |
| **USE CASE 3** | Use the system to create a capital city report. |
| **Goal in Context** | Someone has asked a worker to create a report about Belgium. They need to display the name, country and population. |
| **Scope and Level** | Company, Summary. |
| **Preconditions** | We know the capital city and have the system ready to use and connected to the MYSQL database. |
| **Success End Condition** | Worker returns a report about Belgium. |
| **Failed End Condition** | Worker was unable to return report. |
| **Primary, Secondary Actors** | Worker, database. |
| **Trigger** | Someone asking for the capital city report. |
| **Description** | 1. Company asks worker to get report of Belgium. 2. Worker uses the system to return the capital cities data columns by entering its name into the return query. 3. Worker generates the report for the company to use. |

|  |  |
| --- | --- |
| **USE CASE 4** | Use the system to create a population report. |
| **Goal in Context** | A worker needs to produce a population report of India. They need to display the name, the total population, the population living in cities and the population not living in cities. |
| **Scope & Level** | Company, Summary. |
| **Preconditions** | The list of continents/regions/countries is already available and known to us in the system. |
| **Success End Condition** | The worker receives a generated report of the population with accurate information. |
| **Failed End Condition** | The worker does not receive a population report and is left with no information. |
| **Primary, Secondary Actors** | Worker, Database. |
| **Trigger** | A need for a population report. |
| **Description** | 1. Worker is asked to get a report of a specific continent/region/country population. 2. Worker enters continent/region/country name into system. 3. Worker generates report of the specified location’s population from the system. |

|  |  |
| --- | --- |
| **USE CASE 5** | Use the system to create a report where part of the data returned is sorted from largest to smallest. |
| **Goal in Context** | A worker needs to create a report to show all the countries in the world organized by population largest to smallest, so anyone that uses the report can clearly see the differences in population displayed. |
| **Scope & Level** | Company, Summary. |
| **Preconditions** | We know which populations we want to find, and we have the system ready to use and connected to the MYSQL database. |
| **Success End Condition** | The worker generates a report of the population with accurate information. |
| **Failed End Condition** | The worker does not receive a population report or receives a population report with unsorted information. |
| **Primary, Secondary Actors** | Worker, Database. |
| **Trigger** | Someone needs to view an ordered list of the population for a given location. |
| **Description** | 1. Worker is asked to get the country report, and to sort this report by the population figure ordered largest to smallest. 2. Worker creates a country report in the system and specifies to sort this information from largest to smallest by writing a SQL statement with the ‘ORDER BY’ command performed on the population. 3. Worker generates country report with the population sorted from largest to smallest from the system. |

|  |  |
| --- | --- |
| **USE CASE 6** | Use the system to create a report where the number of results returned is entered by the user. |
| **Goal in Context** | A worker needs to create a report to show only the top 5 populated countries in the world. |
| **Scope & Level** | Company, Summary. |
| **Preconditions** | We know the specific number of entries we want to return, and we have the system ready to use and connected to the MYSQL database. |
| **Success End Condition** | The worker generates a report which shows only the top 5 populated countries. |
| **Failed End Condition** | The worker does not receive a country report or receives a report which shows < or > 5 results. Or we are not shown the top 5 populations and instead only shown the first 5 results. |
| **Primary, Secondary Actors** | Worker, Database. |
| **Trigger** | Someone needs to see only the top n populated countries |
| **Description** | 1. Worker is asked to get a report of the top 5 populated countries. 2. Worker enters the country name into the system and uses SQL ‘ORDER BY’ command to sort the information by population. Then use SQL ‘LIMIT 5’ command to only return the top 5 results. 3. Worker generates a report which shows the top 5 populated countries. |

|  |  |
| --- | --- |
| **USE CASE 7** | Use the system to create a report to show multiple population figures. |
| **Goal in Context** | A worker needs to create a report to show the population of people, people living in a city and people not living in a city in Spain. |
| **Scope & Level** | Company, Summary. |
| **Preconditions** | We know the name of the continent we want to look at, and we have the system ready to use and connected to the MYSQL database. |
| **Success End Condition** | The worker generates a report to show the population of a country, along with how many of those people live in a city or not. |
| **Failed End Condition** | The worker returns no result, the worker does not return all 3 criteria. For example, we only get the total population of Spain. |
| **Primary, Secondary Actors** | Worker, Database. |
| **Trigger** | Someone needs to see the living trends of the people in a given continent. |
| **Description** | 1. Worker is asked to show the population of people, people living in cities and people not living in cities in a specific country. 2. Worker enters the country name into the system and uses a SQL statement to group all people that live in one of the cities. The people not living in a city will be the total population – people in cities. 3. Worker generates a report which shows the 3 different population figures. |

|  |  |
| --- | --- |
| **USE CASE 8** | Use the system to create a report to show the population of the world and of a specified continent, region, country, district and city. |
| **Goal in Context** | A worker needs to see the population of Japan for use in their work. |
| **Scope & Level** | Company, Summary. |
| **Preconditions** | We know the name of the location’s population we want, and we have the system ready to use and connected to the MYSQL database. |
| **Success End Condition** | The worker returns the population of Japan. |
| **Failed End Condition** | The worker returns no result. |
| **Primary, Secondary Actors** | Worker, Database. |
| **Trigger** | Someone needs to see the population of a given location. |
| **Description** | 1. Worker needs to see Japan’s population. 2. Worker enters the countries name and uses an SQL statement to return only the population. 3. Worker returns the value for Japan’s population. |

|  |  |
| --- | --- |
| **USE CASE 9** | Use the system to create a report to show the number of people that speak a specified language, sorted by population largest to smallest, and show it as a percentage of the world population |
| **Goal in Context** | A worker needs detailed information about the number of people that speak English, to use in their work. |
| **Scope & Level** | Company, Summary. |
| **Preconditions** | We know the language we want information about, and we have the system ready to use and connected to the MYSQL database. |
| **Success End Condition** | The worker returns the number of people that speak English, sorted from largest to smallest, and shown as a percentage of the world’s population. |
| **Failed End Condition** | The worker returns no result or returns the number of people that speak English unsorted, or returns the number of people without showing as a percentage. |
| **Primary, Secondary Actors** | Worker, Database. |
| **Trigger** | Worker needs to create a report about a specified language. |
| **Description** | 1. Worker needs to see information about the people that speak English. 2. Worker creates a SQL statement to return the population that speaks English, then use the SQL ‘ORDER BY’ command on the population figure to sort this information from largest to smallest. Save this number. 3. Then use basic arithmetic operators to find the percentage. ‘speakEnglish/total world population x 100’. 4. Worker generates report which shows the relevant information. |