1. **SELECT**

* **SPEC:** Employees can view the names of other employees working in the system.

**QUERY:** Output a list of employees

**SQL:** **SELECT** Name FROM Employee

|  |
| --- |
| NAME |
| Lebron |
| Kobe |
| Michael |
| Pau |
| Kevin |
| Rajon |
| Deron |
| Mario |
| Dwight |
| Larry |

1. **DISTINCT**

* **SPEC:** The system must be able to show the list of cities where employees are from.

**QUERY:** Display all the distinct cities where the employees are from

**SQL:** SELECT **DISTINCT** city FROM Employee

|  |
| --- |
| CITY |
| Aurora |
| Brampton |
| Kingston |
| Scarborough |
| Toronto |
| Vaughan |

1. **BETWEEN**

* **SPEC:** The system should be able to display ticket types from a specific range of price

**QUERY:** output a list of ticket types with a price range between 3 and 6

**SQL:** SELECT Type FROM Tickets WHERE Price **BETWEEN** 3 AND 6

|  |
| --- |
| TYPE |
| Child |
| Student |
| Senior |

1. **NOT**

* **SPEC:** The system should be able to find all the employees who are not from a specific city.

**QUERY:** output a list of employees who lives in a city other than Toronto

**SQL:** SELECT Name FROM Employee WHERE city **NOT** IN(‘Toronto’)

|  |
| --- |
| NAME |
| Kobe |
| Kevin |
| Rajon |
| Deron |
| Mario |
| Dwight |
| Larry |

1. **SUM**

* **SPEC:** Keep track of total amount of inventory items.

**Query:** Get the total number of products in the inventory.

**SQL:** SELECT **SUM(Quantity)** As Total\_Products FROM Inventory

|  |
| --- |
| Total\_Products |
| 2656 |

**6)** **LESS THAN** ( < )

* **SPEC:** Find the employees who work for minimum wage, where minimum wage is less than 15

**Query:** Find the names of the employees who earn less than 15.

**SQL:** SELECT Name FROM Employee WHERE Wage **<** 15

|  |
| --- |
| NAME |
| Kevin |
| Dwight |

1. **IN**

* **SPEC:** The system should display a set of employees and their ids from multiple cities.

**Query:** Find the names of all the employees and their Id’s who are located in Toronto or Scarborough.

**SQL:** SELECT Name, ID FROM Employee WHERE city **IN(‘Toronto’, ‘Scarborough’)**

|  |  |
| --- | --- |
| NAME | ID |
| Lebron | 5003 |
| Michael | 5005 |
| Pau | 5006 |
| Kevin | 5007 |
| Mario | 5010 |

1. **WHERE**

* **SPEC:** The system must be able to display all employees and ids according to a specified job title.

**QUERY:** output a list of employees with their ids with the title “Cleaner”

**SQL:** SELECT ID, Name FROM Employees **WHERE** Title=’Cleaner’

|  |  |
| --- | --- |
| ID | NAME |
| 5007 | Kevin |
| 5008 | Rajon |
| 5011 | Dwight |

1. **ORDER BY**

* **SPEC:** The system must be able to display all employees according to id

**QUERY:** Output a list of employees with their cities and their ids sorted in ascending order

**SQL:** SELECT Id, Name, City FROM Employee **ORDER BY** Id

|  |  |  |
| --- | --- | --- |
| ID | NAME | CITY |
| 5003 | Lebron | Toronto |
| 5004 | Kobe | Brampton |
| 5005 | Michael | Toronto |
| 5006 | Pau | Toronto |
| 5007 | Kevin | Scarborough |
| 5008 | Rajon | Vaughan |
| 5009 | Deron | Brampton |
| 5010 | Mario | Scarborough |
| 5011 | Dwight | Kingston |
| 5012 | Larry | Aurora |

1. **UNION**

* **SPEC:** The system must find employees according to their job title or where they live

**QUERY:** output a list of employees whose title is Manager or lives in Toronto

**SQL:** SELECT Name FROM Employee WHERE Title=’Manager’ **UNION** SELECT Name FROM Employee WHERE City=’Toronto’

|  |
| --- |
| NAME |
| Lebron |
| Michael |
| Pau |

1. **ALL**

* **SPEC:** The system must be able to identify the employee with the highest pay rate

**QUERY:** Find the employee who has the highest wage

**SQL:** SELECT Name FROM Employee WHERE Wage >= **ALL(SELECT wage FROM Employee)**

|  |
| --- |
| NAME |
| Larry |

1. **AND**

* **SPEC:** The system should show the quantity in inventory and the tank location of a specified fish

**Query:** Find the tank id and its quantity in which the Catfish is placed in

**SQL:** SELECT Tank.Id, Inventory.Quantity FROM Tank, Inventory WHERE product=’Catfish’ **AND** Inventory.Name = ‘Catfish’

|  |  |
| --- | --- |
| ID | QUANTITY |
| 2421 | 300 |



* **SPEC:** The system must identify the employee responsible for maintaining a specified tank

**Query:** Find the employee name who cleans Tank Id 2181

**SQL:** SELECT Employee.Name FROM Employee, Maintenance WHERE TankID=2181 AND Employee.ID=Maintenance.EmployeeID

|  |
| --- |
| NAME |
| Michael |

**14)**

* **SPEC:** The system must display entrance fees or all prices after taxes.

**QUERY:** Output a list of prices with 50% increase in each ticket price

**SQL:** SELECT Type, Price \* 1.5 FROM Tickets

|  |  |
| --- | --- |
| TYPE | Price |
| Child | 6.0 |
| Student | 7.5 |
| Adult | 13.5 |
| Senior | 9.0 |

1. **COUNT**

* **SPEC:** Given the fish name and tank id, the system must be able to determine the number of fishes that are in the tank.

**Query:** Find the number of Goldfishes in tank 5345 and display the quantity available in total

**SQL:** SELECT **COUNT(Product)** AS Goldfishes FROM Tank WHERE Tank.ID=5345

|  |
| --- |
| Goldfishes |
| 1 |

1. **NULL**

* **SPEC:** Find all the fishes in the inventory that exist in the system

**Query:** Display the list of fishes in the inventory and is not NULL

**SQL:** SELECT Name FROM Inventory WHERE ID IS NOT **NULL**

|  |
| --- |
| NAME |
| Catfish |
| Amango |
| Angelfish |
| Barfish |
| Carpsucker |
| Blowfish |
| FlatLoach |
| Goldfish |
| Grideye |
| Kelpfish |

1. **LIKE %**

* **SPEC:** Asearch for employees with a given First letter or last letter

**Query:** Display a list of name of products that start with ‘A’ and ends with ‘o’

**SQL:** SELECT Name FROM Inventory WHERE (Name **LIKE** ‘A**%**’) OR (NAME **LIKE** ‘%o’)

1. **LIKE \_**

* **SPEC:** A search for employees with a given initial

**Query:** Display a list of employee names that start with ‘C’

**SQL:** SELECT Name FROM Employee WHERE (Name **LIKE** ‘C**\_**’)

1. **ANY**

* **SPEC:** Find the employees who have a higher wage than the employees from a specific department

**Query:** Display a list of employees and their phone numbers whose wage is greater than some wage who is in Sales.

**SQL:** SELECT Name, Phone FROM Employee WHERE Wage > **ANY** (SELECT Wage FROM Employee WHERE Title=’Sales’)

|  |  |
| --- | --- |
| NAME | PHONE |
| Kobe | 657472 |
| Michael | 456123 |
| Larry | 785473 |

1. **ORDER BY DESC**

* **SPEC:** To see which inventory item is close to zero, the system must keep track of fishes whose quantity are in descending order

**Query:** Output the list of fish names with quantities in descending order.

**SQL:** SELECT Id, Name, Quantity FROM Inventory **ORDER BY** Quantity **DESC**

|  |  |  |
| --- | --- | --- |
| ID | NAME | QUANTITY |
| 546584 | Goldfish | 546 |
| 698544 | Kelpfish | 349 |
| 397243 | Catfish | 300 |
| 897659 | Carpsucker | 289 |
| 423424 | Barfish | 256 |
| 298021 | Blowfish | 232 |
| 379922 | Grideye | 213 |
| 203811 | FlatLoach | 192 |
| 108233 | Angelfish | 156 |
| 108201 | Amango | 123 |

1. **VIEW**

* **SPEC:** For privacy purposes, system should show ONLY the names of employees whose wages are above average.

**Query:** Create a view that displays the employee names who have a higher than average pay.

**SQL 1):** CREATE **VIEW** greaterThanAvgPay AS SELECT Name FROM Employee WHERE Wage > (SELECT AVG(Wage) FROM Employee)

**SQL 2):** Select Name AS Employees\_With\_Greater\_Than\_Avg\_Pay FROMgreaterThanAvgPay

|  |
| --- |
| Employees\_With\_Greater\_Than\_Avg\_Pay |
| Kobe |
| Michael |
| Pau |
| Deron |
| Larry |

1. **ASSERTION**

* **SPEC:** Employees cannot be assigned a wage lower than minimum.

**Query:** Create an assertion that checks if wages are greater than 10.

**SQL:** CREATE **ASSERTION** constraintWages CHECK (SELECT Wage FROM Employee WHERE Wage > 10)

1. **TRIGGER**

* **SPEC:** the system must be able to replace new products when old one runs out

**Query:** Add 50 more fishes when quantity of fish runs out.

**SQL:** CREATE TRIGGER updateQuantity AFTER UPDATE OF quantity ON inventory FOR EACH ROW UPDATE inventory SET quantity=50 WHERE quantity <= 0

**24)**

**- SPEC:** Only cleaners are allowed to clean tank

**Query:** Create an assertion that checks to see if only employees who are cleaners can clean a tank.

**SQL:** CREATE **ASSERTION** cleaners CHECK (SELECT maintenance.employeeid from maintenance where maintenance.employeeid = employee.id AND employee.id = ‘Cleaner’)

**25) EXISTS**

**- SPEC:** The system must be able to find a fish and its quantity if it exists.

**Query:** Find Shark if it exists in the system. If it does, find the quantity available in the system.

**SQL:** Select name, quantity from inventory where **EXISTS**(select name from inventory where name=’shark’)

**26) GROUP BY**

**- SPEC:** They system should be able to count all employees located in the workplace for each department.

**Query:** Find the number of employees in each department

**SQL:** SELECT Title, COUNT(title) AS Number\_In\_Dept FROM employee **GROUP BY** title

|  |  |
| --- | --- |
| TITLE | NUMBER\_IN\_DEPT |
| Cleaner | 3 |
| CustRep | 2 |
| Guide | 1 |
| Sales | 2 |
| Stock | 1 |

**27) AVG(column name)**

**- SPEC:** The system should be able to find the average wages of all employees in each department

**Query:** Find the average wages for each department

**SQL:** SELECT title, **AVG(wage)** AS Avg\_Pay FROM employee GROUP BY title

|  |  |
| --- | --- |
| TITLE | AVG\_PAY |
| Cleaner | 15 |
| CustRep | 22 |
| Guide | 16 |
| Sales | 20 |
| Stock | 19 |

**28) MAX (column name)**

**- SPEC:** The system should display the wage of the greatest paid employee for each department

**Query:** Find the greatest wage of each department

**SQL:** SELECT title, **Max(wage)** AS Greatest\_Pay FROM employee GROUP BY title

|  |  |
| --- | --- |
| TITLE | GREATEST\_PAY |
| Cleaner | 15 |
| Guide | 16 |
| Stock | 19 |

**29) AS**

* **SPEC:** Keep track of which employees are cleaning tanks and the number of employees cleaning the tank.

**Query:** Find the amount of employees cleaning each tank

**SQL:** Select tankid, COUNT(tankid) **AS** Number\_of\_employees from maintenance group by tankid

|  |  |
| --- | --- |
| TANKID | NUMBER\_OF\_EMPLOYEES |
| 2108 | 1 |
| 2181 | 1 |
| 2415 | 1 |
| 2421 | 1 |
| 4367 | 1 |
| 4935 | 1 |
| 5345 | 1 |
| 5457 | 1 |
| 9654 | 1 |
| 9847 | 1 |

**30) HAVING**

**- SPEC:** Keep track of employees who are not cleaning tanks

**Query:** Display all the tanks that do not have cleaners assigned to them

**SQL:** SELECT tankid, COUNT(tankid) ASNumber\_of\_employees FROM maintenance GROUP BY tankid **HAVING** COUNT(tankid) = 0

|  |  |
| --- | --- |
| TANKID | NUMBER\_OF\_EMPLOYEES |
|  |  |

**31) Dot**

**- SPEC:** Keep track of employees who are cleaners and which tank they are maintaining

**Query:** Find which employees have tanks assigned to them for cleaning if they are cleaners

**SQL:** SELECT employee.id, employee.name FROM employee, maintenance WHERE employee.id=maintenance.employeeid AND employee.title='Cleaner'

|  |  |
| --- | --- |
| ID | NAME |
| 5007 | Kevin |
| 5008 | Rajon |
| 5011 | Dwight |

**32) EXCEPT**

**- SPEC:** Keep track of the fishes that are not displayed in the tanks but are in the inventory.

**Query:** Find all the fishes that are not in tanks but are in inventory

**SQL:** SELECT name FROM Inventory

**EXCEPT**

SELECT product FROM Tank

|  |
| --- |
| 1 |
| Shark |

**33)**

* **SPEC:** Show ONLY the manager’s name and phone number to customers.

**QUERY:** Create a view for customers that shows the Manager’s name and phone number

**SQL:** CREATE VIEW managerinfo AS SELECT name, phone FROM Employee WHERE title=’Manager’

SELECT name,phone FROM managerinfo

|  |  |
| --- | --- |
| NAME | PHONE |
| 5012 | 785473 |

**34) OR**

**- SPEC:**

**QUERY:** Find the id and name of employee with id 5011 or name Larry

**SQL:** SELECT id, name FROM Employee WHERE id=5011 **OR** name=’Larry’

|  |  |
| --- | --- |
| EID | ENAME |
| 5011 | Dwight |
| 5012 | Larry |

**35) MIN(column name)**

**- SPEC:** The system should display the wage of a lowest paid employee within a specific department

**QUERY:** Find the lowest paid employee who is a Cleaner

**SQL:** SELECT Min(wage) AS Lowest\_Wage\_From\_Dept From employee WHERE Title=’Cleaner’

|  |
| --- |
| Lowest\_Wage\_From\_Dept |
| 13 |

**36)**

**SPEC:** Keep track of tanks that do not have fishes

**Query:** List all tank ids without fishes

**SQL:** SELECT DISTINCT tid AS tanks\_without\_fishes FROM Tank WHERE Tank.pname IS NULL OR Tank.pname=''

|  |
| --- |
| TANKS\_WITHOUT\_FISHES |
|  |

**37)**

**SPEC:** Find the profit/loss after selling X tickets of each type after employee wage expenses.

**Query:** Show profit or loss after selling 142 tickets of each type.

**SQL:** SELECT (SUM(price \* 142)) - SUM(wage\*8\*4) AS profit\_loss FROM Employee, Tickets

|  |
| --- |
| PROFIT\_LOSS |
| 10784 |

**38)**

**SPEC:** Keep track of employee’s total earnings for the month, assuming each employee works 8 hours a day, 4 weeks per month.

**Query:** Show each employee’s name and total earnings for the month.

**SQL:** SELECT ename, wage\*4\*8 AS employee\_total\_earnings\_$ FROM Employee

|  |  |
| --- | --- |
| ENAME | EMPLOYEE\_TOTAL\_EARNINGS\_$ |
| Lebron | 576 |
| Kobe | 704 |
| Micheal | 672 |
| Paul | 608 |
| Kevin | 448 |
| Rajon | 512 |
| Deron | 608 |
| Dwight | 384 |
| Larry | 832 |