DB

Certainly! Here's the provided SQL code written in Beautiful Obsidian Markdown:

```
# Lab_5_1
## Use ITI

### 1. Retrieve number of students who have a value in their age.
```sql
SELECT COUNT(St_id)
FROM Student
WHERE St_age IS NOT NULL
```

2. Get all instructors Names without repetition

```sql
SELECT DISTINCT ins_name
FROM Instructor
```

3. Display student with the following Format (use isNull function)

```sql
SELECT
    S.St_Id AS [Student ID],
    ISNULL(S.St_Fname + ' ' +S.St_Lname,'No Name') AS [Student Full Name],
    ISNULL(D.Dept_name,'No Department') AS [Department Name]
FROM Student AS S
JOIN Department AS D
ON S.Dept_id = D.Dept_id
```

4. Display instructor Name and Department Name

```sql
SELECT I.Ins_Name, D.Dept_Name
FROM Instructor AS I
LEFT OUTER JOIN Department AS D
ON I.Dept_Id = D.Dept_Id
```

5. Display student full name and the name of the course he is taking

```sql
SELECT
    ISNULL(S.St_Fname + ' ' +S.St_Lname,'No Name') AS [Student Full Name],
    C.Crs_Name AS [Course Name]
FROM Student AS S
```
```

```
JOIN Stud_Course AS SC
ON S.St_Id = SC.St_Id
JOIN Course AS C
ON SC.Crs_ID = C.Crs_Id
WHERE SC.Grade IS NOT NULL
```

6. Display number of courses for each topic name

```
SELECT T.Top_Name, COUNT(C.crs_id) AS [Number of Courses]
FROM Topic AS T
JOIN Course AS C
ON T.Top_Id = C.Top_Id
GROUP BY T.Top_Name
```

7. Display max and min salary for instructors

```
SELECT MAX(salary) AS [Max Salary], MIN(salary) AS [Min Salary]
FROM Instructor
WHERE Salary IS NOT NULL
```

8. Display instructors who have salaries less than the average salary of all instructors.

```
SELECT Ins_Name
FROM Instructor
WHERE Salary < (SELECT AVG(salary) FROM Instructor)
```

9. Display the Department name that contains the instructor who receives the minimum salary.

```
-- Sol 1 Using SubQuery
SELECT D.Dept_Name
FROM Instructor AS I
JOIN Department AS D
ON I.Dept_Id = D.Dept_Id
WHERE I.salary = (SELECT MIN(salary) FROM Instructor)

-- Sol 2 Using top
SELECT D.Dept_Name

FROM Department AS D
WHERE D.Dept_Id = (
    SELECT TOP (1) I.Dept_Id
    FROM Instructor AS I
```

```
    ORDER BY I.salary
)

-- Sol 3 Using Ranking
SELECT Dept_Name
FROM (
    SELECT Dept_Name, ROW_NUMBER() OVER (ORDER BY I.salary) AS RowNum
    FROM Instructor AS I
    JOIN Department AS D
    ON I.Dept_Id = D.Dept_Id
) AS Ranking_Sub_Query
WHERE RowNum = 1
```

10. Select max two salaries in instructor table.

```
-- OR
SELECT TOP (2) Salary, ROW_NUMBER() OVER (ORDER BY Salary DESC) AS RowNum
FROM Instructor
```

11. Select instructor name and his salary but if there is no salary display instructor bonus keyword. “use coalesce Function”

```
SELECT Ins_Name, COALESCE(CONVERT(VARCHAR(10), Salary), 'Bonus')
FROM Instructor
```

12. Select Average Salary for instructors

```
SELECT AVG(salary)
FROM Instructor
```

13. Select Student first name and the data of his supervision

```
SELECT S.St_Fname, Sup.*
FROM Student AS S
JOIN Student AS Sup
ON S.St_Id = Sup.St_super
```

14. Write a query to select the highest two salaries in Each Department for instructors who have salaries. “using one of Ranking Functions”

```sql
SELECT Dept_name, Salary
FROM (
    SELECT Dept_name, Salary, ROW_NUMBER() OVER (PARTITION BY D.dept_id ORDER BY I.salary
DESC) AS RowNum
    FROM Instructor AS I
    JOIN Department AS D
    ON I.Dept_Id = D.Dept_id
    WHERE Salary IS NOT NULL
) AS [Ranking Query]
WHERE RowNum IN (1,2)
```

15. Write a query to select a random student from each department. “using one of Ranking Functions”

```sql
SELECT TOP(1) *
FROM Student
ORDER BY NEWID()
```

Lab_5_2

Use AdventureWorks2012

1. Display the SalesOrderID, ShipDate of the SalesOrderHeader table (Sales schema) to show SalesOrders that occurred within the period ‘7/28/2002’ and ‘7/29/2014’

```sql
SELECT SalesOrderID, ShipDate
FROM Sales.SalesOrderHeader
WHERE OrderDate BETWEEN '7/28/2002' AND '7/29/2014'
```

2. Display only Products(Production schema) with a StandardCost below $110.00 (show ProductID, Name only)

```sql
SELECT ProductID, Name
FROM Production.Product
WHERE StandardCost < 110
```

3. Display ProductID, Name if its weight is unknown

```sql
SELECT ProductID, Name
FROM Production.Product
```

```
WHERE Weight IS NULL
```

4. Display all Products with a Silver, Black, or Red Color

```
SELECT ProductID, Name
FROM Production.Product
WHERE color IN ('silver', 'black', 'red')
```

5. Display any Product with a Name starting with the letter B

```
SELECT *
FROM Production.Product
WHERE Name LIKE ('B%')
```

6. Run the following Query

```
-- UPDATE Production.ProductDescription
-- SET Description = 'Chromoly steel_High of defects'
-- WHERE ProductDescriptionID = 3
UPDATE Production.ProductDescription
SET Description = 'Chromoly steel_High of defects'
WHERE ProductDescriptionID = 3
```

7. Write a query that displays any Product description with an underscore value in its description.

```
SELECT Description
FROM Production.ProductDescription
WHERE Description LIKE '%[_]%'
```

8. Calculate sum of TotalDue for each OrderDate in Sales.SalesOrderHeader table for the period between '7/1/2001' and '7/31/2014'

```
SELECT SUM(TotalDue)
FROM Sales.SalesOrderHeader
WHERE OrderDate BETWEEN '7/1/2001' AND '7/31/2014'
```

9. Display the Employees HireDate (note no repeated values are allowed)

```sql
SELECT DISTINCT HireDate
FROM HumanResources.Employee
```

10. Calculate the average of the unique ListPrices in the Product table

```sql
SELECT AVG(DISTINCT ListPrice)
FROM Production.Product
```

11. Display the Product Name and

its ListPrice within the values of 100 and 120 the list should has the following format "The [product name] is only! [List price]" (the list will be sorted according to its ListPrice value)

```sql
SELECT 'The' + Name + ' is only! ' + CONVERT(VARCHAR(50), ListPrice) AS Offers
FROM Production.Product
WHERE ListPrice BETWEEN 100 AND 120
ORDER BY ListPrice
```

12. Transfer the rowguid, Name, SalesPersonID, Demographics from Sales.Store table in a newly created table named [store_Archive]

```sql
SELECT rowguid, Name, SalesPersonID, Demographics
INTO Sales.store_Archive
FROM Sales.Store
```

13. Try the previous query but without transferring the data?

```sql
SELECT rowguid, Name, SalesPersonID, Demographics
INTO Sales.store_Archive
FROM Sales.Store
WHERE 1 = 2
```

14. Using union statement, retrieve today’s date in different styles using convert or format function.

```sql
SELECT FORMAT(GETDATE(), 'd-M-yyyy')
UNION
SELECT FORMAT(GETDATE(), 'ddd-MMM-yyyy')
```

```
UNION
SELECT FORMAT(GETDATE(), 'dddd-MMMM-yyyy')
```