Name: Nguyen Trung Hieu

Databse test



**Câu 1**: Giải thích các hàm thông dụng sau

|  |  |  |
| --- | --- | --- |
| Stt | Tên Hàm | Mục đích sử dụng & nên sử dụng khi nào |
| 1 | Count() | Hàm COUNT() trả về số lượng dòng phù hợp với một tiêu chí được chỉ định. |
| 2 | Sum() | Hàm SUM() trả về tổng của một cột số. |
| 3 | MAX() | Hàm MAX()trả về giá trị lớn nhất trong một cột số |
| 4 | MIN() | Hàm MIN()trả về giá trị nhỏ nhất trong một cột số. |
| 5 | NVL() | Hàm NVL() cho phép thay thế một giá trị null bằng một giá trị khác. |
| 6 | TO\_CHAR() | Hàm TO\_CHAR() dùng để chuyển đổi một số hoặc ngày tháng sang kiểu chuỗi. |
| 7 | T0\_DATE() | Hàm TO\_DATE() có tác dụng chuyển định dạng của một chuỗi hoặc một dãy số sang định dạng ngày tháng |
| 8 | TO\_NUMBER() | Hàm TO\_NUMBER() dùng để chuyển đổi một chuỗi sang kiểu số |
| 9 | SUBSTR() | Hàm SUBSTR() dùng để trích xuất một chuỗi con từ một chuỗi |
| 10 | REPLACE() | Hàm REPLACE() dùng để thay thế chuỗi ký tự này thành chuỗi ký tự khác trong một chuỗi cho trước |
| 11 | REVERSE() | Hàm REVERSE() dung để đảo ngược một chuỗi |
| 12 | DECODE() | Hàm DECODE() dùng để hỗ trợ tính năng câu lệnh IF-THEN-ELSE hay CASE |
| 13 | TRUNC() | Hàm TRUNC() dùng để thu gọn một số đến một vị trí nào đó sau dấu phẩy |
| 14 | LENGTH() | Hàm LENGTH() dùng để đếm tổng số ký tự có trong chuỗi, kể cả ký tự khoảng trắng. |
| 15 | lPAD() | Hàm LPAD() trả về một chuỗi được đệm vào bên trái với một chuỗi được chỉ định theo một độ dài nhất định. |
| 16 | RPAD() | Hàm RPAD() trả về một chuỗi được đệm vào bên phải với một chuỗi được chỉ định theo một độ dài nhất định. |
| 17 | TRIM() | Hàm **TRIM() dung để** loại bỏ các khoảng trắng 2 bên của một string. Ta có thể chỉ định các ký tự để loại bỏ. |
| 18 | LTRIM() | Hàm L**TRIM() dùng để** loại bỏ các khoảng trắng bên trái của một string. Ta có thể chỉ định các ký tự để loại bỏ. |
| 19 | RTRIM() | Hàm R**TRIM() dùng để** loại bỏ các khoảng trắng bên phải của một string. Ta có thể chỉ định các ký tự để loại bỏ. |
| 20 | ROUND() | Hàm ROUND() dùng để làm tròn số đến một vị trí phần nguyên hoặc vị trí phần thập phân được chỉ định. |
| 21 | ADD\_MONTHS() | Hàm ADD\_MONTHS() có tác dụng thêm hoặc bớt đi một hoặc nhiều tháng từ date truyền vào. |

**Câu 2**:

SELECT \*

FROM MDM\_CUSTOMER A

WHERE A.CUST\_LGL\_ENG\_NM LIKE ‘%\\_LOGISTICS%’ ESCAPE ‘\’

ORDER BY A.CUST\_LOCL\_LANG\_NM NULLS FIRST

1. Vui lòng giải thích ý nghĩa của câu SQL trên

* Lấy ra các giá trị của bảng MDM\_CUSTOMER theo điều kiện giá trị cột CUST\_LGL\_ENG\_NM phải chứa chuỗi “\_LOGISTICS” , sau đó sắp xếp theo cột CUST\_LOCL\_LANG theo thứ tự null được xếp đầu.

1. ý nghĩa của việc dùng ESCAPSE

* Để xóa ký tự ‘\’ , với mục đích biến ký tự ‘\_’ trở về 1 ký tự bình thường

C) Ý nghĩa của việc dung Nulls First.

* Dùng để đưa những thuộc tính có giá trị null lên đầu.

D) Ý nghĩa của việc dung alias, có nên dung alias trong mọi trường hợp không?

* Dùng ALIAS giúp cho câu SQL ngắn gọn, dễ nhìn hơn, đỡ nhầm lẫn khi các cột cùng tên, tiết kiệm được thời gian với những thuộc tính có tên dài hoặc được gọi nhiều.ều
* Không nên, chỉ nên dùng trong các trường hợp cụ thể.

**Câu 3**:

SELECT \*

FROM MDM\_CUSTOMER

WHERE 1 = 1

AND CUST\_CNT\_CD = 'JP'

AND TO\_CHAR(CUST\_SEQ) = '201791'

Theo bạn câu trên cách dung TO\_CHAR(CUST\_SEQ) = '201791' có hợp lý không, tại sao?

* Không hợp lý, bởi vì oracle có hỗ trợ so sánh số với một chuỗi số, vậy nên việc dùng TO\_CHAR sẽ dư thừa và làm giảm tốc độ xử lý.

**Câu 4**: cho câu SQL và kết quả như hình bên dưới



1. Giải thích ý nghĩa COUNT(\*), COUNT(1), COUNT(CUST\_LOCL\_LANG\_NM)

* COUNT(\*) : đếm số lượng dòng trả về, kể cả giá trị null
* COUNT(1) : đếm số lượng dòng trả về, kể cả giá trị null
* COUNT(CUST\_LOCL\_LANG\_NM): đếm số lượng dòng trả về, chỉ tính những dòng có giá trị khác null ở thuộc tính CUST\_LOCL\_LANG\_NM

1. Tại sao COUNT(CUST\_LOCL\_LANG\_NM) lại bằng 0

* Tại vì các dòng trả về, giá trị của thuộc tính CUST\_LOCL\_LANG\_NM đều bằng null

**Câu 5**: có 2 cách như bên dưới, cách nào tốt tại sao

|  |  |  |
| --- | --- | --- |
|  | **Cách 1** | **Cách 2** |
|  | NVL(SUM(COL1),0) | SUM(NVL(COL1,0)) |

* Cách 1 sẽ tối ưu hơn, vì hàm NVL chỉ chạy 1 lần. trong khi cách 2 hàm NVL phải chạy n lần.

**Câu 6**: có 2 cách như bên dưới, cách nào tốt tại sao

|  |  |  |
| --- | --- | --- |
|  | **Cách 1** | **Cách 2** |
|  | * NVL(SUM(COL1),0) + NVL(SUM(COL2),0) | * Ex.1] SUM(NVL(COL1 + COL2,0)) * Ex.2] NVL(SUM(COL1 + COL2),0) |

* Cách 1 sẽ tốt hơn, vì cách 2 nếu như kết quả 1 trong 2 cột bị null thì kết quả tổng sẽ null theo.

**Câu 7**: có 2 cách như bên dưới, cách nào tốt tại sao

|  |  |  |
| --- | --- | --- |
|  | **Cách 1** | **Cách 2** |
|  | SELECT A.CUST\_NO, A.ORD\_NO, A.PRO\_CD, B.PROD\_NM  FROM TB\_ORD A,  TB\_PROD B  WHERE 1 = 1  AND A.PRO\_CD = B.PROD\_CD  AND B.PROD\_CD IN (SELECT PROD\_CD FROM TB\_PROD D WHERE D.PROD\_CD = A.PRO\_CD AND PROD\_UNIT\_AMT < 800); | SELECT A.CUST\_NO, A.ORD\_NO, A.PRO\_CD, B.PROD\_NM  FROM TB\_ORD A,  TB\_PROD B  WHERE 1 = 1  AND A.PRO\_CD = B.PROD\_CD  AND EXISTS (SELECT D.PROD\_CD FROM TB\_PROD D WHERE D.PROD\_CD = A.PRO\_CD AND D.PROD\_UNIT\_AMT < 800); |

**IN** đc dung khi nào và EXISTS đc dung khi nào.

Cách 1 tốt hơn vì kết quả truy vấn con ít, vì vậy hiệu năng của IN sẽ nhanh hơn EXISTS

* IN được dùng trong trường hợp truy vấn con có kết quả truy vấn nhỏ, không cần dữ liệu từ bảng khác. IN còn có thể được dùng truy vấn giá trị
* EXISTS được dùng trong trường hợp truy vấn con có kết quả truy vấn lớn, cần dữ liệu tư bảng khác. EXISTS chỉ có thể dùng trong truy vấn con

**Câu 8**: có 2 cách như bên dưới, cách nào tốt tại sao

|  |  |  |
| --- | --- | --- |
|  | **Cách 1** | **Cách 2** |
|  | SELECT A.CUST\_NO, A.ORD\_NO, A.PRO\_CD, B.PROD\_NM  FROM TB\_ORD A,  TB\_PROD B  WHERE 1 = 1  AND A.PRO\_CD = B.PROD\_CD  AND B.PROD\_CD IN ('00001','00002'); | SELECT A.CUST\_NO, A.ORD\_NO, A.PRO\_CD  , (SELECT B.PROD\_NM FROM TB\_PROD B WHERE B.PROD\_CD = A.PRO\_CD) AS PROD\_NM  FROM TB\_ORD A  WHERE 1 = 1  AND A.PRO\_CD IN ('00001','00002'); |

* Cách số 1 sẽ tốt hơn, bởi vì khi xử lí data với số lượng ít, sử dụng phé join sẽ hoạt động tốt hơn subquery.

**Câu 9**: cho số 8988.80 vui lòng xuất ra định dạng $8,988.800

SELECT TO\_CHAR(8988.80, '$9,999.999') FROM DUAL

**Câu 10**: cho số 8988.80, 820988.80 vui lòng xuất ra định dạng $8,000.000, $820,000.000

SELECT TO\_CHAR(TRUNC(8988.80,-3),'$9,999.99')FROM DUAL

SELECT TO\_CHAR(TRUNC(820988.80 ,-3),'$999,999.99')FROM DUAL

**Câu 11**: Cho cấu SQL và kết quả như sau:



Như hình trên cả 2 A và B điêu substr từ 1, đến 3 tại sao kết quả lại khác nhau.

* Khi dùng TO\_CHAR để chuyển sang kiểu string, ta sẽ có thêm 1 ký tự ở đầu đại diện cho dấu, nhưng vì số ta truyền vào là dương, nên ký tự đầu sẽ là khoảng trắng.
* Mà A có fm ở tham số format\_mask nên sẽ bị xóa đi khoảng trắng, còn B thì không.
* Vậy nên subtring của B lấy 3 ký tự đầu, sẽ bao gồm có khoảng trắng ở ký tự đầu

**Câu 12**: Viết Câu SQL xuất ra, Ngày hiện tại, này hôm qua, ngày mai

SELECT sysdate-1 as Yesterday, sysdate as Today ,sysdate+1 as Tomorrow from dual

**Câu 13**: ta có table (**TB\_ORD**), yêu cầu viết câu SQL để generate ORD\_NO có đô dài 10 tự với format sau: yyyymmdd000Seq, ví dụ hnay là 20191028 và chưa có seq nào thì ORD\_NO sẽ là 201910280001, và nếu đã tồn tại ORD\_NO 201910280001 thì nó sẽ là 201910280002

INSERT INTO TB\_ORD(CUST\_NO, ORD\_NO, PRO\_CD, ORD\_DTTM)

SELECT :CUST\_NO,:ORD\_NO,:PRO\_CD ,

DECODE(COUNT(\*), 0, RPAD(TO\_CHAR(SYSDATE, 'YYYYMMDD'), 12, '0') + 1, MAX(ORD\_DTTM)+1) AS DTM

FROM TB\_ORD

WHERE ORD\_DTTM LIKE TO\_CHAR(SYSDATE, 'YYYYMMDD')||'%'

**Câu 14**: ta có table (**MDM\_CUSTOMER**) và dữ liệu như bên dưới



Các field liên quan: CUST\_CNT\_CD, CUST\_SEQ, CUST\_GRP\_HRCHY\_CD, CUST\_GRP\_ID

Dữ liệu cột CUST\_GRP\_HRCHY\_CD có thể có(**I**: individual, **C**: Country, **G**: Global)

1. Viết câu SQL tìm CUST\_GRP\_ID sao cho: CUST\_GRP\_HRCHY\_CD có **I** hoặc **C** nhưng không có **G**

SELECT DISTINCT(CUST\_GRP\_ID)

FROM MDM\_CUSTOMER

WHERE CUST\_GRP\_ID NOT IN (SELECT CUST\_GRP\_ID

FROM MDM\_CUSTOMER

WHERE CUST\_GRP\_HRCHY\_CD = 'G')

AND CUST\_GRP\_HRCHY\_CD IN('I', 'C');

1. Viết câu SQL tìm CUST\_GRP\_ID sao cho: CUST\_GRP\_HRCHY\_CD có **G** và có **I** nhưng không có **C**

SELECT DISTINCT(CUST\_GRP\_ID)

FROM MDM\_CUSTOMER A

WHERE CUST\_GRP\_ID NOT IN (SELECT CUST\_GRP\_ID

FROM MDM\_CUSTOMER

WHERE CUST\_GRP\_HRCHY\_CD = 'C')

AND EXISTS(SELECT CUST\_GRP\_ID FROM MDM\_CUSTOMER B

WHERE

A.CUST\_GRP\_HRCHY\_CD like 'G'

AND B.CUST\_GRP\_HRCHY\_CD like 'I'

AND A.CUST\_GRP\_ID = B.CUST\_GRP\_ID)

**Câu 15**: ta có table (**TB\_PROD**) và dữ liệu như bên dưới



Viets cấu SQL để suất ra kêt quả như sau:

1. Lấy max(PROD\_UNIT\_AMT)
2. Lấy giá trị min(PROD\_UNIT\_AMT)
3. Lấy giá trị trung bình PROD\_UNIT\_AMT
4. Lấy tên của sản phẩm có PROD\_UNIT\_AMT lớn nhất

Kết quả phải ra đc như sau:



SELECT A.MAX\_AMT , A.MIN\_AMT , A.AVG, B.PROD\_NM AS MAX\_NAME

FROM (SELECT MAX(PROD\_UNIT\_AMT) AS MAX\_AMT,

MIN(PROD\_UNIT\_AMT) AS MIN\_AMT,

AVG(PROD\_UNIT\_AMT) AS AVG

FROM TB\_PROD) A, TB\_PROD B

WHERE B.PROD\_UNIT\_AMT = A.MAX\_AMT

**Câu 16**: ta có table (**TB\_ORD**) và dữ liệu như bên dưới



A) viết cấu SQL lấy ra top3 sản phẩm đc bán nhiều nhất**.**

SELECT CUST\_NO,SUM(PRO\_CD) AS TOTAL

FROM TB\_ORD

GROUP BY CUST\_NO

ORDER BY(TOTAL) DESC

FETCH FIRST 3 ROWS ONLY;

B) Viết cấu SQL lấy ra cái ORD\_DT, ORD\_TM, PROD\_CD gần nhất theo CUST\_NO

Kết quả mong đợi như sau:



SELECT \*

FROM (

SELECT CUST\_NO,ORD\_DTTM, ROW\_NUMBER () OVER (PARTITION BY CUST\_NO ORDER BY ORD\_DTTM DESC) rn

FROM TB\_ORD

GROUP BY CUST\_NO, ORD\_DTTM

) A

WHERE A.RN = 1;

SELECT A.DT, B.PRO\_CD, NVL(B.TOTAL,0) AS TOTAL

FROM

(

SELECT '201906' AS DT FROM DUAL

UNION ALL

SELECT '201907' AS DT FROM DUAL

UNION ALL

SELECT '201908' AS DT FROM DUAL

UNION ALL

SELECT '201909' AS DT FROM DUAL

) A

LEFT OUTER JOIN

(SELECT B.PRO\_CD, SUBSTR(B.ORD\_DTTM, 1, 6) AS ORD\_DTTM, COUNT(\*)

AS TOTAL

FROM TB\_ORD B

GROUP BY B.PRO\_CD, SUBSTR(B.ORD\_DTTM, 1, 6)

) B PARTITION BY (B.PRO\_CD)

ON A.DT = B.ORD\_DTT;

C) viết cấu SQL report xem trong tháng 06, 07, 08, 09 cảu 2019 sản phẩm có mã code là 00001bán đc bao nhiêu cái.

Kết quả mong đợi nhưu sau: left outer join partition - 130



SELECT \*

FROM (

SELECT CUST\_NO,ORD\_DTTM, ROW\_NUMBER () OVER (PARTITION BY CUST\_NO ORDER BY ORD\_DTTM DESC) rn

FROM TB\_ORD

GROUP BY CUST\_NO, ORD\_DTTM

) A

WHERE A.RN = 1;

D) giả sư lúc đầu sản phẩn 00001 có 100 cái, viết report để tính số lương remain theo tháng 06, 07, 08, 09



SELECT A.DT, B.PRO\_CD, NVL(B.TOTAL,0) AS TOTAL, 100 - NVL(SUM(B.TOTAL) OVER (PARTITION BY B.PRO\_CD ORDER BY A.DT),0) AS REMAIN

FROM

(

SELECT '201906' AS DT FROM DUAL

UNION ALL

SELECT '201907' AS DT FROM DUAL

UNION ALL

SELECT '201908' AS DT FROM DUAL

UNION ALL

SELECT '201909' AS DT FROM DUAL

) A

LEFT OUTER JOIN

(SELECT B.PRO\_CD, SUBSTR(B.ORD\_DTTM, 1, 6) AS ORD\_DTTM, COUNT(\*)

AS TOTAL

FROM TB\_ORD B

WHERE B.PRO\_CD = '00001'

GROUP BY B.PRO\_CD, SUBSTR(B.ORD\_DTTM, 1, 6)

) B PARTITION BY (B.PRO\_CD)

ON A.DT = B.ORD\_DTTM;