Database test



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

|  |  |  |
| --- | --- | --- |
| Stt | Tên Hàm | Mục đích sử dụng & nên sử dụng khi nào |
| 1 | Count() | * COUNT([ALL | DISTINCT | \* ] expression): Là hàm tổng hợp trả về số dòng trong một nhóm * Trả về số dòng (số lượng) phù hợp với một tiêu chí cụ thể. * Nên dùng khi cần đếm số lần xuất hiện của một tiêu chí nào đó, hoặc check tồn tại của một đối tượng. |
| 2 | Sum() | * SUM([ALL | DISTINCT] expression): Là hàm trả về tổng của một tập các giá trị (không tính NULL value) |
| 3 | MAX() | * MAX(expression): Là hàm trả về giá trị lớn nhất của một tập các giá trị (không tính NULL value). |
| 4 | MIN() | * MIN(expression) Tìm giá trị nhỏ nhất trên cột |
| 5 | NVL() | * NVL(e1, e2): Cho phép ta thay thế một giá trị NULL thành một giá trị khác Hàm NVL này nhận vào 2 tham số e1, e2. Nếu e1 là NULL, thì hàm này sẽ trả về e2, ngược lại thì trả về e1 Hai tham số e1 và e2 có thể có cùng kiểu hoặc khác kiểu dữ liệu. Nếu khác kiểu thì sẽ được chuyển đổi dựa trên các quy tắc sau: * Nếu e1 là kiểu ký tự thì e2 sẽ được chuyển đổi thành kiểu dữ liệu của e1 trước khi so sánh với NULL |
| 6 | TO\_CHAR() | * Chuyển đổi một giá trị bất kỳ sang kiểu String |
| 7 | T0\_DATE() | * TO\_CHAR(expr [, date\_format] [, nslparam]): Là hàm chuyển đổi dữ liệu có kiểu DATE hoặc INTERVAL thành dạng string có format được chỉ định |
| 8 | TO\_NUMBER() | * TO\_NUMBER( string1 [, format\_mask] [, nls\_language] ): có tác dụng chuyển đổi string thành số |
| 9 | SUBSTR() | * SUBSTR( str, start\_position [, substring\_length] ): Lấy ra chuỗi phụ từ chuỗi lớn |
| 10 | REPLACE() | * REPLACE(string\_expression, string\_pattern [,string\_replacement]): Thay thế một ký tự/chuỗi bằng một chuỗi khác |
| 11 | REVERSE() | * REPLACE(string\_expression, string\_pattern [,string\_replacement]): Đảo ngược chuỗi ký tự |
| 12 | DECODE() | * DECODE( expression, search, result [, search, result]... [, default] ): Bản chất là IF – THEN – ELSE function, hay case when |
| 13 | TRUNC() | * TRUNC( number [, decimal\_places] ): Thu gọn một số đến một vị trí nào đó sau dấu phẩy (vị trí được chỉ định bởi tham số truyền vào) |
| 14 | LENGTH() | * TRUNC( number [, decimal\_places] ): Trả về dộ dài của chuỗi được truyền vào |
| 15 | LPAD() | * LPAD(source\_string, target\_length [,pad\_string]): Dùng  để đệm vào bên trái của kết quả chuỗi với tập ký tự được xác định trong tham số. |
| 16 | RPAD() | * RPAD(source\_string, target\_length [,pad\_string]): Dùng  để đệm vào bên phải của kết quả chuỗi với tập ký tự được xác định trong tham số. |
| 17 | TRIM() | * TRIM( [ [ LEADING | TRAILING | BOTH ] trim\_character FROM ] trim\_source): Loại bỏ các ký tự giống ký tự được chỉ định từ đầu hoặc cuối của chuỗi |
| 18 | LTRIM() | * LTRIM(trim\_source,[set]): Loại bỏ các ký tự giống với ký tự được chỉ định từ phía bên trái của chuỗi |
| 19 | RTRIM() | * RTRIM(trim\_source,[set]): Loại bỏ các ký tự giống với ký tự được chỉ định từ phía bên phải của chuỗi |
| 20 | ROUND() | * ROUND( number [, decimal\_places] ): Trả về một số được làm tròn đến chữ số thập phân được chọn |
| 21 | ADD\_MONTHS() | * ADD\_MONTHS(date\_expression, month): Trả về một date được sau khi được cộng vào số tháng được chọn (có thể cộng số âm) |

**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**

A) Vui lòng giải thích ý nghĩa của câu SQL trên:

- Hiển thị tất cả các cột trong bảng mdm\_customer theo điều kiện và sắp theo cột locl\_lang\_nm với các giá trị null ở các dòng đầu tiên.

B) ý nghĩa của việc dùng ESCAPSE:

- Biến ký tự sau escape trở thành ký tự bình thường

C) Ý nghĩa của việc dung Nulls First:

- Để những ô null lên đầu tiên

D) Ý nghĩa của việc dung alias, có nên dung alias trong mọi trường hợp không?  
- Giúp đặt tên ngắn, dễ viết.

- Không nên dùng alias cho mọi trường hợp

**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 dùng TO\_CHAR(CUST\_SEQ) = '201791' có hợp lý không, tại sao?

* Không hợp lý. Mặc dù kết quả có thể đúng nhưng sẽ mất index.

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



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

- Count(\*): đếm tổng số hàng trong bảng, bao gồm các giá trị null

- Count(1): gán giá trị từ dấu ngoặc đơn cho mọi hàng trong bảng rồi đếm đếm tổng số hàng trong bảng, bao gồm các giá trị null

- Count(CUST\_LOCL\_LANG\_NM): Đếm tất cả các hàng trong cột được chỉ định không bao gồm giá trị null

B) Tại sao COUNT(CUST\_LOCL\_LANG\_NM) lại bằng 0:

- Vì tất cả các hàng đều 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 tốt hơn. Vì cách 1 chạy nhanh hơn

**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 2 tốt hơn. Vì cách 2 chạy nhanh hơn

**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); |

* Cách 1 tốt hơn vì cách 1 cost thấp hơn
* Dùng IN khi kết quả truy vấn phụ nhỏ, dùng EXISTS khi kết quả truy vấn phụ lớn

**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 2 tốt hơn. Vì cách 2 chạy nhanh hơn (cùng cost)

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

**SELECT** TO\_CHAR (8988.80,

'$9,9999.999')

**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), 'L9G999G999D00')

**FROM** dual;

**SELECT** to\_char(trunc(820988.80, -3), 'L9G999G999D00')

**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.

* Vì độ dài của to\_char(98765,’fm00000’) là 5 số và độ dài của to\_char(98765,’00000’) là 6 số. Do fm loại 1 khoảng trắng dành để biểu diễn dấu + - trong 1 chuỗi số trả về.

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

**SELECT** sysdate + interval '-1' **DAY** **AS** YESTERDAY,

sysdate **AS** NOW,

sysdate + interval '1' **DAY** **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

**SELECT** concat(to\_char(SYSDATE, 'yyyyMMdd'),

(**SELECT** to\_char(**count**(\*) + 1, 'fm0000')

**FROM** tb\_ord

**WHERE** ord\_dttm **like** concat(to\_char(SYSDATE, 'yyyyMMdd'), '\_\_\_\_')))

**FROM** tb\_ord

**WHERE** ROWNUM = 1

**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)

A) 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 **like** 'G')

B) 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

**WHERE** CUST\_GRP\_ID **not** **in**

(**SELECT** CUST\_GRP\_ID

**FROM** mdm\_customer

**WHERE** cust\_grp\_hrchy\_cd **like** 'C')

**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)

**SELECT** **max**(PROD\_UNIT\_AMT)

**FROM** tb\_prod;

1. Lấy giá trị min(PROD\_UNIT\_AMT)

**SELECT** **min**(PROD\_UNIT\_AMT)

**FROM** tb\_prod;

1. Lấy giá trị trung bình PROD\_UNIT\_AMT

**SELECT** **avg**(PROD\_UNIT\_AMT)

**FROM** tb\_prod;

1. Lấy tên của sản phẩm có PROD\_UNIT\_AMT lớn nhất

Kết quả phải ra đc như sau:



**SELECT** prod\_unit\_amt **AS** MAX\_AMT,

prod\_nm **AS** MAX\_NAME,

a.minp **AS** MIN\_AMT,

a.avgp **AS** **AVG**

**FROM** tb\_prod,

(**SELECT** **max**(PROD\_UNIT\_AMT) **AS** MAXP,

**min**(PROD\_UNIT\_AMT) **AS** MINP,

**avg**(PROD\_UNIT\_AMT) **AS** AVGP

**FROM** tb\_prod) A

**WHERE** prod\_unit\_amt =

(**SELECT** **max**(PROD\_UNIT\_AMT)

**FROM** tb\_prod)

**AND** tb\_prod.prod\_unit\_amt = a.maxp;

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



A) viết cấu SQL lấy ra top 3 sản phẩm đc bán nhiều nhất**.**

**SELECT** \*

**FROM**

(**SELECT** pro\_cd,

DENSE\_RANK() OVER(

**ORDER** **BY** **COUNT**(\*) **DESC**) **AS** rank

**FROM** tb\_ord

**GROUP** **BY** pro\_cd) TEMP

**WHERE** temp.rank <= 3;

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** a.cust\_no,

b.ord\_no,

b.ord\_dttm,

b.pro\_cd

**FROM**

(**SELECT** cust\_no,

**MAX**(ord\_dttm) maxp

**FROM** tb\_ord

**GROUP** **BY** cust\_no) A

**LEFT** **JOIN** tb\_ord B **ON** a.cust\_no = b.cust\_no

**AND** a.maxp = b.ord\_dttm

**ORDER** **BY** a.cust\_no;

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:



**SELECT** A.MON,

NVL(B.pro\_cd, '00001') **AS** PRO\_CD,

NVL(b.total, 0)

**FROM**

(**SELECT** **distinct**(substr(ord\_dttm, 1, 6)) **AS** MON

**FROM** tb\_ord

**WHERE** substr(ord\_dttm, 1, 5) **like** '20190'

**AND** substr(ord\_dttm, 6, 1) **in** ('6',

'7',

'8',

'9')) A

**LEFT** **OUTER** **JOIN**

(**SELECT** A.MON,

pro\_cd,

**count**(pro\_cd) **AS** total

**FROM**

(**SELECT** substr(ord\_dttm, 1, 6) **AS** MON,

pro\_cd

**FROM** tb\_ord

**WHERE** substr(ord\_dttm, 1, 5) **like** '20190'

**AND** substr(ord\_dttm, 6, 1) **in** ('6',

'7',

'8',

'9')

**AND** pro\_cd **like** '00001') A

**GROUP** **BY** MON,

pro\_cd) B **ON** A.MON = B.MON;

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** **c**.mon,

**c**.pro\_cd,

**c**.total,

100- NVL(**SUM**(**c**.total) OVER (PARTITION **BY** **c**.pro\_cd

**ORDER** **BY** **c**.mon),0) **AS** REMAIN

**FROM**

(**SELECT** A.MON,

NVL(B.pro\_cd, '00001') **AS** PRO\_CD,

NVL(b.total, 0) **AS** TOTAL

**FROM**

(**SELECT** **distinct**(substr(ord\_dttm, 1, 6)) **AS** MON

**FROM** tb\_ord

**WHERE** substr(ord\_dttm, 1, 5) **like** '20190'

**AND** substr(ord\_dttm, 6, 1) **in** ('6',

'7',

'8',

'9')) A

**LEFT** **OUTER** **JOIN**

(**SELECT** A.MON,

pro\_cd,

**count**(pro\_cd) **AS** total

**FROM**

(**SELECT** substr(ord\_dttm, 1, 6) **AS** MON,

pro\_cd

**FROM** tb\_ord

**WHERE** substr(ord\_dttm, 1, 5) **like** '20190'

**AND** substr(ord\_dttm, 6, 1) **in** ('6',

'7',

'8',

'9')

**AND** pro\_cd **like** '00001') A

**GROUP** **BY** MON,

pro\_cd) B **ON** A.MON = B.MON) **C ;**