Funciones analíticas STDDEV, STDDEV_POP y STDDEV_SAMP

by admin - domingo, octubre 04, 2020

https://dbandtech.com/funciones-analiticas-stddev-stddev_pop-y-stddev_samp/

Este artículo ofrece una descripción general de las funciones analíticas **STDDEV, STDDEV_POP y STDDEV_SAMP**. Si es nuevo en las funciones analíticas, probablemente debería leer primero esta introducción a las funciones analíticas.

Preparar

Los ejemplos de este artículo requieren la siguiente tabla.

```
--DROP TABLE emp PURGE;

CREATE TABLE emp (
  empno    NUMBER(4) CONSTRAINT pk_emp PRIMARY KEY,
  ename    VARCHAR2(10),
  job      VARCHAR2(9),
  mgr      NUMBER(4),
  hiredate DATE,
  sal      NUMBER(7,2),
  comm     NUMBER(7,2),
  deptno   NUMBER(2)
);

INSERT INTO emp VALUES (7369,'SMITH','CLERK',7902,to_date('17-12-1980'
,'dd-mm-yyyy'),800,NULL,20);
INSERT INTO emp VALUES (7499,'ALLEN','SALESMAN',7698,to_date('20-2-198
1','dd-mm-yyyy'),1600,300,30);
INSERT INTO emp VALUES (7521,'WARD','SALESMAN',7698,to_date('22-2-1981
','dd-mm-yyyy'),1250,500,30);
INSERT INTO emp VALUES (7566,'JONES','MANAGER',7839,to_date('2-4-1981'
,'dd-mm-yyyy'),2975,NULL,20);
INSERT INTO emp VALUES (7654,'MARTIN','SALESMAN',7698,to_date('28-9-19
81','dd-mm-yyyy'),1250,1400,30);
```

```
INSERT INTO emp VALUES (7698,'BLAKE','MANAGER',7839,to_date('1-5-1981'
,'dd-mm-yyyy'),2850,NULL,30);
INSERT INTO emp VALUES (7782,'CLARK','MANAGER',7839,to_date('9-6-1981'
,'dd-mm-yyyy'),2450,NULL,10);
INSERT INTO emp VALUES (7788,'SCOTT','ANALYST',7566,to_date('13-JUL-87
','dd-mm-rr')-85,3000,NULL,20);
INSERT INTO emp VALUES (7839,'KING','PRESIDENT',NULL,to_date('17-11-19
81','dd-mm-yyyy'),5000,NULL,10);
INSERT INTO emp VALUES (7844,'TURNER','SALESMAN',7698,to_date('8-9-198
1','dd-mm-yyyy'),1500,0,30);
INSERT INTO emp VALUES (7876,'ADAMS','CLERK',7788,to_date('13-JUL-87',
 'dd-mm-rr')-51,1100,NULL,20);
INSERT INTO emp VALUES (7900,'JAMES','CLERK',7698,to_date('3-12-1981',
'dd-mm-yyyy'),950,NULL,30);
INSERT INTO emp VALUES (7902,'FORD','ANALYST',7566,to_date('3-12-1981'
,'dd-mm-yyyy'),3000,NULL,20);
INSERT INTO emp VALUES (7934,'MILLER','CLERK',7782,to_date('23-1-1982'
,'dd-mm-yyyy'),1300,NULL,10);
COMMIT;
```

STDDEV, STDDEV_POP y STDDEV_SAMP como funciones agregadas

Las **funciones agregadas STDDEV, STDDEV_POP y STDDEV_SAMP** se utilizan para calcular la desviación estándar, la desviación estándar de la población y la desviación estándar de la muestra acumulada de un conjunto de datos, respectivamente. Como funciones agregadas, reducen el número de filas, de ahí el término "**aggregate**". Si los datos no están agrupados, convertimos las 14 filas de la tabla EMP en una sola fila con los valores agregados.

```
SELECT STDDEV(sal) AS stddev_sal,
       STDDEV_POP(sal) AS stddev_pop_sal,
       STDDEV_SAMP(sal) AS stddev_samp_sal
FROM   emp;

STDDEV_SAL STDDEV_POP_SAL STDDEV_SAMP_SAL
---------- -------------- ---------------
1182.50322     1139.48862      1182.50322

SQL>
```

Podemos obtener más granularidad de la información al incluir una cláusula **GROUP BY**. En el siguiente ejemplo, vemos los valores por departamento.

```
SELECT deptno,
       STDDEV(sal) AS stddev_sal,
       STDDEV_POP(sal) AS stddev_pop_sal,
       STDDEV_SAMP(sal) AS stddev_samp_sal
FROM   emp
GROUP BY deptno
ORDER BY deptno;

    DEPTNO STDDEV_SAL STDDEV_POP_SAL STDDEV_SAMP_SAL
---------- ---------- -------------- ---------------
        10 1893.62967     1546.14215      1893.62967
        20  1123.3321     1004.73877       1123.3321
        30 668.331255     610.100174      668.331255

SQL>
```

En ambos casos, hemos agregado los datos para obtener los valores, devolviendo menos filas de las que comenzamos. Las funciones analíticas nos permiten devolver estos valores agregados conservando los datos de la fila original.

Función analítica STDDEV

Si hay más de un registro en la muestra después de descartar valores nulos, la función **STDDEV** devuelve el resultado de la función **STDDEV_SAMP**, la desviación estándar acumulativa de la muestra. Si solo hay una fila en la muestra después de descartar los valores nulos, la función **STDDEV** devuelve el valor "0". Si no hay registros en el conjunto después de descartar los valores nulos, el valor devuelto es **NULL**.

La descripción básica de la función analítica STDDEV se muestra a continuación.

```
STDDEV([ DISTINCT | ALL ] expr) [ OVER (analytic_clause) ]
```

El uso de una cláusula **OVER** vacía convierte la función **STDDEV** en una función analítica. La falta de una cláusula de partición significa que todo el conjunto de resultados se trata como una sola partición, por lo que obtenemos la desviación estándar del salario de todos los empleados, así como todos los datos originales.

```
SELECT empno,
       ename,
       deptno,
       sal,
       STDDEV(sal) OVER () AS stddev_sal
FROM   emp
```

```
ORDER BY deptno;

     EMPNO ENAME          DEPTNO        SAL STDDEV_SAL
---------- ---------- ---------- ---------- ----------
      7782 CLARK              10       2450 1182.50322
      7839 KING               10       5000 1182.50322
      7934 MILLER             10       1300 1182.50322
      7566 JONES              20       2975 1182.50322
      7902 FORD               20       3000 1182.50322
      7876 ADAMS              20       1100 1182.50322
      7369 SMITH              20        800 1182.50322
      7788 SCOTT              20       3000 1182.50322
      7521 WARD               30       1250 1182.50322
      7844 TURNER             30       1500 1182.50322
      7499 ALLEN              30       1600 1182.50322
      7900 JAMES              30        950 1182.50322
      7698 BLAKE              30       2850 1182.50322
      7654 MARTIN             30       1250 1182.50322

SQL>
```

Agregar la cláusula de partición nos permite mostrar la desviación estándar del salario por departamento, junto con los datos de los empleados para cada departamento.

```
SELECT empno,
       ename,
       deptno,
       sal,
       STDDEV(sal) OVER (PARTITION BY deptno) AS stddev_sal_by_dept
FROM   emp;

     EMPNO ENAME          DEPTNO        SAL STDDEV_SAL_BY_DEPT
---------- ---------- ---------- ---------- ------------------
      7782 CLARK              10       2450         1893.62967
      7839 KING               10       5000         1893.62967
      7934 MILLER             10       1300         1893.62967
      7566 JONES              20       2975          1123.3321
      7902 FORD               20       3000          1123.3321
      7876 ADAMS              20       1100          1123.3321
      7369 SMITH              20        800          1123.3321
      7788 SCOTT              20       3000          1123.3321
      7521 WARD               30       1250         668.331255
      7844 TURNER             30       1500         668.331255
      7499 ALLEN              30       1600         668.331255
```

```
      7900 JAMES              30         950         668.331255
      7698 BLAKE              30        2850         668.331255
      7654 MARTIN             30        1250         668.331255

SQL>
```

Función analítica STDDEV_POP

La función **STDDEV_POP** devuelve la desviación estándar de la población, la raíz cuadrada de la función **VAR_POP**.
La descripción básica de la función analítica STDDEV_POP se muestra a continuación.

```
STDDEV_POP(expr) [ OVER (analytic_clause) ]
```

El uso de una cláusula **OVER** vacía convierte la función **STDDEV_POP** en una función analítica. La falta de una cláusula de partición significa que todo el conjunto de resultados se trata como una sola partición, por lo que obtenemos la desviación estándar de población del salario de todos los empleados, así como todos los datos originales.

```
SELECT empno,
       ename,
       deptno,
       sal,
       STDDEV_POP(sal) OVER () AS stddev_pop_sal
FROM   emp;

     EMPNO ENAME          DEPTNO        SAL STDDEV_POP_SAL
---------- ---------- ---------- ---------- --------------
      7369 SMITH              20        800     1139.48862
      7499 ALLEN              30       1600     1139.48862
      7521 WARD               30       1250     1139.48862
      7566 JONES              20       2975     1139.48862
      7654 MARTIN             30       1250     1139.48862
      7698 BLAKE              30       2850     1139.48862
      7782 CLARK              10       2450     1139.48862
      7788 SCOTT              20       3000     1139.48862
      7839 KING               10       5000     1139.48862
      7844 TURNER             30       1500     1139.48862
      7876 ADAMS              20       1100     1139.48862
      7900 JAMES              30        950     1139.48862
      7902 FORD               20       3000     1139.48862
      7934 MILLER             10       1300     1139.48862
```

```
SQL>
```

Agregar la cláusula de partición nos permite mostrar la desviación estándar de población del salario por departamento, junto con los datos de los empleados para cada departamento.

```
SELECT empno,
       ename,
       deptno,
       sal,
       STDDEV_POP(sal) OVER (PARTITION BY deptno) AS stddev_pop_by_dep
t
FROM   emp;

     EMPNO ENAME          DEPTNO        SAL STDDEV_POP_BY_DEPT
---------- ---------- ---------- ---------- ------------------
      7782 CLARK              10       2450         1546.14215
      7839 KING               10       5000         1546.14215
      7934 MILLER             10       1300         1546.14215
      7566 JONES              20       2975         1004.73877
      7902 FORD               20       3000         1004.73877
      7876 ADAMS              20       1100         1004.73877
      7369 SMITH              20        800         1004.73877
      7788 SCOTT              20       3000         1004.73877
      7521 WARD               30       1250         610.100174
      7844 TURNER             30       1500         610.100174
      7499 ALLEN              30       1600         610.100174
      7900 JAMES              30        950         610.100174
      7698 BLAKE              30       2850         610.100174
      7654 MARTIN             30       1250         610.100174

SQL>
```

Función analítica STDDEV_SAMP

La función **STDDEV_SAMP** devuelve la desviación estándar de la muestra acumulada, la raíz cuadrada de la función **VAR_SAMP**.

La descripción básica de la función analítica STDDEV_SAMP se muestra a continuación.

```
STDDEV_SAMP(expr) [ OVER (analytic_clause) ]
```

El uso de una cláusula **OVER** vacía convierte la función **STDDEV_SAMP** en una función analítica. La falta de una cláusula de partición significa que todo el conjunto de resultados se trata como una sola partición, por lo que obtenemos la desviación estándar muestral acumulativa del salario de todos los empleados, así como todos los datos originales.

```
SELECT empno,
       ename,
       deptno,
       sal,
       STDDEV_SAMP(sal) OVER () AS stddev_samp_sal
FROM   emp;

     EMPNO ENAME          DEPTNO        SAL STDDEV_SAMP_SAL
---------- ---------- ---------- ---------- ---------------
      7369 SMITH              20        800      1182.50322
      7499 ALLEN              30       1600      1182.50322
      7521 WARD               30       1250      1182.50322
      7566 JONES              20       2975      1182.50322
      7654 MARTIN             30       1250      1182.50322
      7698 BLAKE              30       2850      1182.50322
      7782 CLARK              10       2450      1182.50322
      7788 SCOTT              20       3000      1182.50322
      7839 KING               10       5000      1182.50322
      7844 TURNER             30       1500      1182.50322
      7876 ADAMS              20       1100      1182.50322
      7900 JAMES              30        950      1182.50322
      7902 FORD               20       3000      1182.50322
      7934 MILLER             10       1300      1182.50322

SQL>
```

Agregar la cláusula de partición nos permite mostrar la desviación estándar muestral acumulada del salario por departamento, junto con los datos de los empleados para cada departamento.

```
SELECT empno,
       ename,
       deptno,
       sal,
       STDDEV_SAMP(sal) OVER (PARTITION BY deptno) AS stddev_samp_by_d
ept
FROM   emp;

     EMPNO ENAME          DEPTNO        SAL STDDEV_SAMP_BY_DEPT
---------- ---------- ---------- ---------- ------------------
```

```
      7782 CLARK          10       2450        1893.62967
      7839 KING           10       5000        1893.62967
      7934 MILLER         10       1300        1893.62967
      7566 JONES          20       2975         1123.3321
      7902 FORD           20       3000         1123.3321
      7876 ADAMS          20       1100         1123.3321
      7369 SMITH          20        800         1123.3321
      7788 SCOTT          20       3000         1123.3321
      7521 WARD           30       1250        668.331255
      7844 TURNER         30       1500        668.331255
      7499 ALLEN          30       1600        668.331255
      7900 JAMES          30        950        668.331255
      7698 BLAKE          30       2850        668.331255
      7654 MARTIN         30       1250        668.331255

SQL>
```

Enlaces rápido
