

關聯式資料庫管理系統 結構化查詢語言 (SQL) 使用 MySQL

資料查詢 - SELECT 命令
子查詢
(Sub-Queries)

子查詢(Sub-Queries)

- ▶ 利用子查詢來解決另類查詢的問題
 - 請找出薪資比MARY高的所有員工資料

主查詢(Main Query)



“哪些員工薪資比MARY的薪資高?”

子查詢(Sub-query)



“Mary的薪資是多少?”



子查詢(Sub-Queries)

- ▶ 主查詢中含有另一個查詢(SELECT)
- ▶ 查詢時先執行子查詢並將結果傳回主查詢

Main-Query

```
SELECT ...  
FROM ...  
WHERE expr operator (SELECT ...  
                        FROM ...  
                        WHERE ... )
```



Sub-Query

- ▶ 子查詢可以出現的位置
 - WHERE 子句
 - HAVING 子句
 - FROM 子句

子查詢(Sub-Queries)

- ▶ 子查詢是巢狀查詢:查詢中有查詢
- ▶ 內部查詢(inner query)的結果會傳給外部查詢(outer query),當成外部查詢運算式中的運算元
- ▶ 依子查詢執行方式分類:
 - 自主子查詢(Self-contained subqueries):與主查詢沒有相關性
 - 相關子查詢(Correlated subqueries):需依賴外部查詢的值
- ▶ 依子查詢結果分類:
 - 純量(scalar)
 - 多值(multi-valued)
 - 多欄位(multiple-column)
 - 表格值(table-valued)

子查詢的型式 (Types of Subqueries)

- ▶ 單筆記錄子查詢 (Single-row subquery)



- ▶ 多筆記錄子查詢 (Multiple-row subquery)



- ▶ 多欄位子查詢 (multiple-column subquery)



- ▶ 表格子查詢 (table-valued subquery)



子查詢(Sub-Queries)

- ▶ 子查詢需用()括住
- ▶ 用在運算子的右邊
- ▶ 子查詢類型&運算子
 - 單筆記錄子查詢(Single Row SubQueries)
 - 單筆記錄運算子(Single Row Operator)
 - >, >=, <, <=, <>
 - 多筆記錄子查詢(Multiple Row SubQueries)
 - 多筆記錄運算子(Multiple Row Operator)
 - IN, ANY, ALL

自主子查詢(Self-contained subqueries)

▶ 自主子查詢


- 子查詢執行與主查詢的資料無關(可獨自測試)

```
SELECT    <select_list>  
FROM      <table>  
WHERE     <expr> operator (SELECT  <select_list>  
                                FROM    <table>);
```

- ▶ 在執行主查詢前，先只執行子查詢一次，再將子查詢的結果傳回給主查詢使用

子查詢(Sub-Queries)- WHERE 子句

- ▶ 列出和JAMES同部門的員工

```
mysql> SELECT empno, ename, deptno
-> FROM emp
-> WHERE deptno =  (SELECT deptno
-> FROM emp
-> WHERE ename = 'JAMES');
```

empno	ename	deptno
7698	BLAKE	30
7654	MARTIN	30
7499	ALLEN	30
7844	TURNER	30
7900	JAMES	30
7521	WARD	30

6 rows in set (0.42 sec)

單筆記錄子查詢(Single Row SubQueries)

- ▶ 列出薪水比員工7566高的所有員工

```
mysql> SELECT empno, ename, sal
-> FROM emp
-> WHERE sal > (SELECT sal
->                FROM emp
->                WHERE empno = 7566);
```

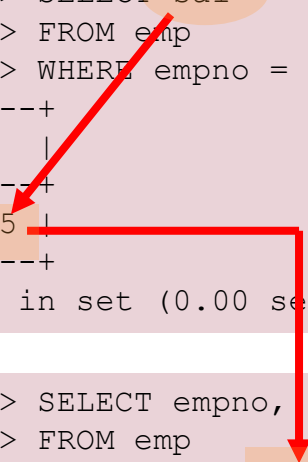
empno	ename	sal
7839	KING	5000
7902	FORD	3000
7788	SCOTT	3000

3 rows in set (0.00 sec)

```
mysql> SELECT sal
-> FROM emp
-> WHERE empno = 7566;
```

sal
2975

1 row in set (0.00 sec)



```
mysql> SELECT empno, ename, sal
-> FROM emp
-> WHERE sal > 2975;
```

empno	ename	sal
7839	KING	5000
7902	FORD	3000
7788	SCOTT	3000

3 rows in set (0.00 sec)

單筆記錄子查詢(Single Row SubQueries)

▶ 多個子查詢在 SELECT 中

```
mysql> SELECT empno, ename, sal, job
-> FROM emp
-> WHERE job = (SELECT job
->                FROM emp
->                WHERE empno = 7499)
-> AND sal > (SELECT sal
->              FROM emp
->              WHERE empno = 7934);
```

SALESMAN


1300

empno	ename	sal	job
7654	MARTIN	1250	SALESMAN
7499	ALLEN	1600	SALESMAN
7844	TURNER	1500	SALESMAN
7521	WARD	1250	SALESMAN

4 rows in set (0.00 sec)

單筆記錄子查詢-使用彙總函數

- ▶ 列出薪資>全公司平均薪資的員工

```
mysql> SELECT empno, ename, sal  
-> FROM emp  
-> WHERE sal >  2073.2143  
-> (SELECT AVG(sal)  
-> FROM emp );
```

empno	ename	sal
7839	KING	5000
7698	BLAKE	2850
7782	CLARK	2450
7566	JONES	2975
7902	FORD	3000
7788	SCOTT	3000

6 rows in set (0.00 sec)

單筆記錄子查詢

► 應用在 HAVING 子句

```
mysql> SELECT deptno, MIN(sal)
-> FROM emp
-> GROUP BY deptno
-> HAVING MIN(sal) >
->
->
->
```

deptno	MIN(sal)
10	1300
30	950

2 rows in set (0.00 sec)



```
(SELECT MIN(sal)
FROM emp
WHERE deptno=20);
```

子查詢

- ▶ 多筆記錄子查詢使用了單一系列運算子

```
mysql> SELECT empno, ename, sal
-> FROM emp
-> WHERE sal =
->             (SELECT MIN(sal)
->             FROM emp
->             GROUP BY deptno);
ERROR 1241 (21000): Subquery returns more than 1 row
```

- ▶ 若子查詢沒傳回資料, 則主查詢傳沒有列傳回

```
mysql> SELECT empno, ename, sal
-> FROM emp
-> WHERE sal > (SELECT sal
->             FROM emp
->             WHERE empno = 8000);
Empty set (0.00 sec)
```

多筆記錄子查詢(Multiple Row SubQueries)

- ▶ 列出公司所有主管

```
mysql> SELECT empno, ename  
-> FROM emp  
-> WHERE empno IN  
-> (SELECT mgr  
-> FROM emp);
```

```
+-----+-----+  
| empno | ename |  
+-----+-----+  
| 7839  | KING  |  
| 7698  | BLAKE |  
| 7782  | CLARK |  
| 7566  | JONES |  
| 7902  | FORD  |  
| 7788  | SCOTT |  
+-----+-----+
```

```
6 rows in set (0.00 sec)
```

NULL 值出現在子查詢中

▶ 列出公司所有職員

```
mysql> SELECT empno, ename  
-> FROM emp  
-> WHERE empno NOT IN  
-> (SELECT mgr  
-> FROM emp);  
Empty set (0.00 sec)
```

- 當子查詢中含有空值且運算子為 not in 時, 主查詢無資料傳回

NULL 值出現在子查詢中

► 解決方法—排除子查詢空值資料

```
mysql> SELECT empno, ename  
-> FROM emp  
-> WHERE empno NOT IN ( SELECT mgr  
-> FROM emp  
-> WHERE mgr IS NOT NULL) ;
```

```
+-----+-----+  
| empno | ename  |  
+-----+-----+  
| 7654  | MARTIN |  
| 7499  | ALLEN  |  
| 7844  | TURNER |  
| 7900  | JAMES  |  
| 7521  | WARD   |  
| 7369  | SMITH  |  
| 7876  | ADAMS  |  
| 7934  | MILLER |  
+-----+-----+  
8 rows in set (0.00 sec)
```

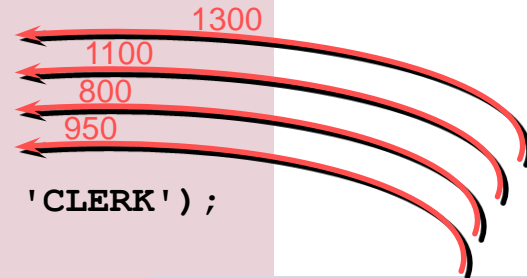

多筆記錄子查詢(Multiple Row SubQueries)

▶ 使用 ANY

```
mysql> SELECT empno, ename, job
-> FROM emp
-> WHERE sal < ANY (SELECT sal
-> FROM emp
-> WHERE job = 'CLERK');
```

empno	ename	job
7369	SMITH	CLERK
7521	WARD	SALESMAN
7654	MARTIN	SALESMAN
7876	ADAMS	CLERK
7900	JAMES	CLERK

5 rows in set (0.00 sec)



```
mysql> SELECT sal
-> FROM emp
-> WHERE job = 'CLERK';
```

sal
800.00
1100.00
950.00
1300.00

4 rows in set (0.00 sec)

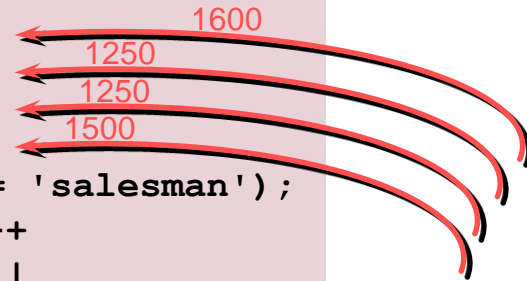
多筆記錄子查詢(Multiple Row SubQueries)

- ▶ 列出薪水比所有salesman高的員工資料

```
mysql> SELECT empno, ename, job, sal
-> FROM emp
-> WHERE sal > ALL (SELECT sal
-> FROM emp
-> WHERE job = 'salesman');
```

empno	ename	job	sal
7566	JONES	MANAGER	2975.00
7698	BLAKE	MANAGER	2850.00
7782	CLARK	MANAGER	2450.00
7788	SCOTT	ANALYST	3000.00
7839	KING	PRESIDENT	5000.00
7902	FORD	ANALYST	3000.00

6 rows in set (0.00 sec)



```
mysql> SELECT sal
-> FROM emp
-> WHERE job = 'salesman';
```

sal
1600.00
1250.00
1250.00
1500.00

4 rows in set (0.00 sec)

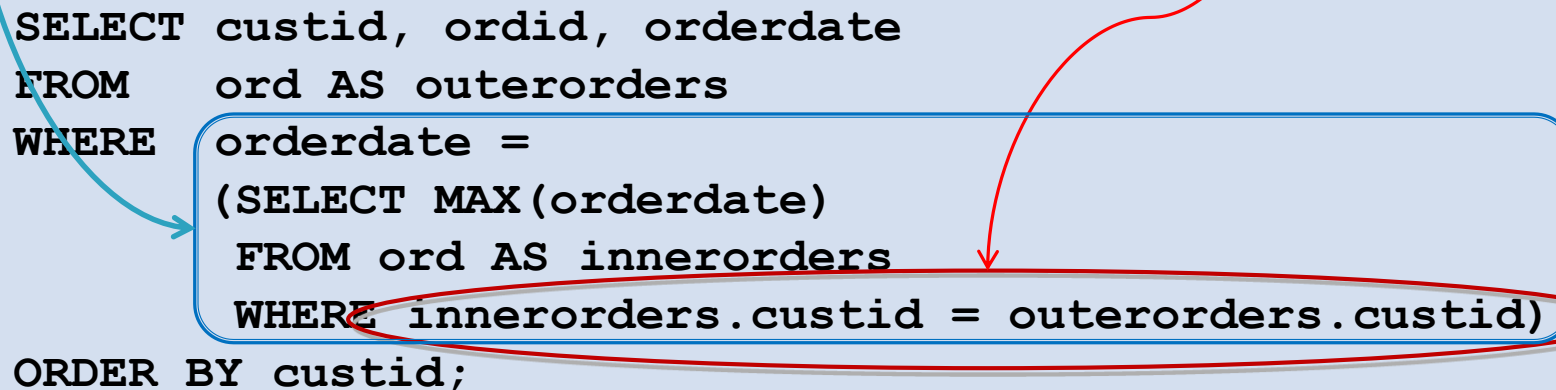
相關子查詢(Correlated subqueries)

- ▶ 相關子查詢必須依賴外部查詢所使用的資料表的資料來執行子查詢
- ▶ 執行次數依主查詢的資料筆數而定
 - 一筆資料執行一次子查詢
- ▶ 可能傳回一個值或多個值
- ▶ 依賴於外部查詢，不能單獨執行
 - 比自主子查詢難測試

撰寫相關子查詢(Correlated Subqueries)

- ▶ 內部查詢如何接收外部查詢資料表的資料
- ▶ 外部查詢如何接受內部查詢執行的結果(scalar or multi-valued)

```
SELECT custid, ordid, orderdate
FROM   ord AS outerorders
WHERE  orderdate =
      (SELECT MAX(orderdate)
       FROM   ord AS innerorders
       WHERE  innerorders.custid = outerorders.custid)
ORDER BY custid;
```



相關子查詢(Correlated subqueries)

- ▶ 查詢每個客戶最近跟公司下訂單的日期資料

```
mysql> SELECT ordid, custid, orderdate
-> FROM      ord AS O1
-> WHERE  orderdate = ( SELECT MAX(orderdate)
->                        FROM      ord AS O2
->                        WHERE  O2.custid = O1.custid)
-> ORDER BY custid, orderdate;
```

ordid	custid	orderdate
613	100	1987-03-12 00:00:00
601	101	1987-01-07 00:00:00
620	102	1987-02-15 00:00:00
616	103	1987-02-03 00:00:00
617	104	1987-02-02 00:00:00
618	105	1987-02-05 00:00:00
607	106	1986-07-14 00:00:00
619	107	1987-02-01 00:00:00
614	108	1987-02-01 00:00:00

9 rows in set (0.05 sec)

相關子查詢(Correlated subqueries)

- ▶ 查詢各部門薪資最高的員工資料

```
mysql> Select ename, sal, deptno  
-> From emp oe  
-> Where sal = (select max(sal)  
->                from emp ie  
->                where ie.deptno=oe.deptno)  
-> Order by deptno;
```

```
+-----+-----+-----+  
| ename | sal      | deptno |  
+-----+-----+-----+  
| KING  | 5000.00  | 10     |  
| SCOTT | 3000.00  | 20     |  
| MARY  | 3000.00  | 20     |  
| BLAKE | 2850.00  | 30     |  
+-----+-----+-----+  
4 rows in set (0.02 sec)
```

使用 EXISTS-存在性測試(existence test)

- ▶ 子查詢可配合使用 EXISTS 來做資料存在性測試(existence test)
 - 只傳回真(True)或假(false)
 - 不會將查詢結果的資料傳回給主查詢
 - 若子查詢有得到任一筆資料則傳回真(True), 若子查詢沒有任何一筆資料則傳回假(false)
- ▶ 語法

```
WHERE [NOT] EXISTS (subquery)
```

使用 EXISTS-存在性測試(existence test)

- ▶ EXISTS 不需任何欄位或運算式
- ▶ 因為不傳回資料, 所以一般在子查詢中的 SELECT 資料項都使用星號 asterisk (*)

```
SELECT  custid, name
FROM    customer AS c
WHERE EXISTS ( SELECT *
                FROM ord AS o
                WHERE c.custid=o.custid);
```

```
SELECT  custid, name
FROM    customer AS c
WHERE NOT EXISTS ( SELECT *
                   FROM ord AS o
                   WHERE c.custid=o.custid);
```


使用 EXISTS-存在性測試(existence test)

- ▶ 列出下過訂單的客戶資料

```
mysql> SELECT  custid, name
-> FROM      customer AS c
-> WHERE EXISTS ( SELECT *
->                FROM ord AS o
->                WHERE c.custid=o.custid) ;
```

custid	name
100	JOCKSPORTS
101	TKB SPORT SHOP
102	VOLLYRITE
103	JUST TENNIS
104	EVERY MOUNTAIN
105	K + T SPORTS
106	SHAPE UP
107	WOMENS SPORTS
108	NORTH WOODS HEALTH AND FITNESS SUPPLY CENTER

9 rows in set (0.05 sec)

使用 NOT EXISTS

- ▶ 列出從未下過訂單的客戶資料

```
mysql> SELECT  custid, name  
-> FROM      customer AS c  
-> WHERE NOT EXISTS ( SELECT *  
->                        FROM ord AS o  
->                        WHERE c.custid=o.custid);  
Empty set (0.00 sec)
```

作業練習

1. 顯示和Blake同部門的所有員工之姓名和進公司日期。
2. 顯示所有在Blake之後進公司的員工之姓名及進公司日期。
3. 顯示薪資比公司平均薪資高的所有員工之員工編號, 姓名和薪資, 並依薪資由高到低排列。
4. 顯示和姓名中包含 T 的人再相同部門工作的所有員工之員工編號和姓名。
5. 顯示在Dallas工作的所有員工之姓名, 部門編號和職稱。
6. 顯示直屬於” King” 的員工之姓名和薪資。
7. 顯示銷售部門” Sales” 所有員工之部門編號, 姓名和職稱。
8. 顯示薪資比公司平均薪資還要高且和名字中有 T 的人在相同部門上班的所有員工之員工編號, 姓名和薪資。
9. 顯示和有賺取佣金的員工之部門編號和薪資都相同的員工之姓名, 部門編號和薪資。
10. 顯示和在Dalls工作的員工之薪資和佣金都相同的員工之姓名, 部門編號和薪資。
11. 顯示薪資和佣金都和Scott相同的所有員工之姓名, 進公司日期和薪資。(不要在結果中顯示Scott的資料)
12. 顯示薪資比所有職稱是” Clerks” 還高的員工之姓名, 進公司日期和薪資, 並將結果依薪資由高至低顯示。