

# Báo cáo Talent 5 Tungdinh – 03.2021

## 1. Báo cáo T0 + 3:

### 1. Backup data

```
SELECT * INTO du_lieu_ban_hang_backup  
FROM du_lieu_ban_hang
```

### 2. Làm sạch dữ liệu

#### 1. Giá trị null: khu\_vuc, thoi\_gian\_trung\_binh và so\_cuoc\_goi

```
DELETE FROM du_lieu_ban_hang_backup WHERE thoi_gian_trung_binh IS NULL
```

```
[75] 1 delete from du_lieu_ban_hang_backup where thoi_gian_trung_binh is null
```

(3 rows affected)

Total execution time: 00:00:01.167

```
DELETE FROM du_lieu_ban_hang_backup WHERE khu_vuc IS NULL
```

```
1 delete from du_lieu_ban_hang_backup where khu_vuc is null
```

(1 row affected)

Total execution time: 00:00:00.173

```
DELETE FROM du_lieu_ban_hang_backup WHERE so_cuoc_goi IS NULL
```

### 2. Dữ liệu trùng:

```
SELECT ngay_ban, san_pham, khu_vuc, so_cuoc_goi, thoi_gian_trung_binh,  
so_luong_ban, camera_truoc, ram, the_nho, dung_luong_pin, man_hinh_rong, don_gia,  
COUNT(*)  
FROM du_lieu_ban_hang_backup  
GROUP BY ngay_ban, san_pham, khu_vuc, so_cuoc_goi, thoi_gian_trung_binh,  
so_luong_ban, camera_truoc, ram, the_nho, dung_luong_pin, man_hinh_rong, don_gia  
HAVING COUNT(*) > 1
```

```
1 select ngay_ban, san_pham, khu_vuc, so_cuoc_goi, thoi_gian_trung_binh, so_luong_ban, camera
2 from du_lieu_ban_hang_backup
3 group by ngay_ban, san_pham, khu_vuc, so_cuoc_goi, thoi_gian_trung_binh, so_luong_ban, camera
4 having count(*) > 1
```

(1 row affected)

Total execution time: 00:00:00.661

	ngay_ban	san_pham	khu_vuc	so_cuoc_goi	thoi_gian_trung_binh	so_luong_ban	camera
1	2018-06-13	BlackBerry KEY2	Bắc Ninh	154	NULL	2126	ε

3. Dữ liệu bất thường: don\_gia của san\_pham VIVO V19 Neo: thay bằng giá trị median 649000

UPDATE du\_lieu\_ban\_hang\_backup SET don\_gia = 6490000 WHERE san\_pham = 'VIVO V19 Neo'  
AND don\_gia = 6500000

```
[90] 1 update du_lieu_ban_hang_backup set don_gia = 6490000 where san_pham = 'VIVO V19 Neo'
2 and don_gia = 6500000
```

(1 row affected)

Total execution time: 00:00:00.170

4. Typo: xảy ra với “bắc ninh” trong khu\_vuc: đổi thành “Bắc Ninh”

UPDATE du\_lieu\_ban\_hang\_backup SET khu\_vuc = N'Bắc Ninh' WHERE khu\_vuc = N'bắc ninh' COLLATE sql\_latin1\_general\_cp1\_cs\_as

```
[127] 1 set khu_vuc = N'Bắc Ninh' where khu_vuc = N'bắc ninh' collate sql_latin1_general_cp1_cs_as
```

(1 row affected)

Total execution time: 00:00:00.279

## 2. Báo cáo T0 + 6

### 1. 1NF

- Mỗi hàng có 1 giá trị atomic: thỏa mãn
- Các cột có tên riêng
- PRIMARY KEY và composite key:  

```
CREATE TABLE du_lieu_ban_hang_1NF (  
ID INT IDENTITY(1,1) PRIMARY KEY,  
ngay_ban DATE,  
san_pham NVARCHAR(100),  
khu_vuc NVARCHAR(100),  
so_cuoc_goi INT,  
thoi_gian_trung_binh INT,  
so_luong_ban INT,  
camera_truoc INT,  
ram INT,  
the_nho INT,  
dung_luong_pin INT,  
man_hinh_rong FLOAT,  
don_gia BIGINT,  
)
```

```
INSERT INTO du_lieu_ban_hang_1NF (ngay_ban, san_pham, khu_vuc, so_cuoc_goi,  
thoi_gian_trung_binh, so_luong_ban, camera_truoc, ram, the_nho, dung_luong_pin,  
man_hinh_rong, don_gia)
```

```
SELECT ngay_ban, san_pham, khu_vuc, so_cuoc_goi, thoi_gian_trung_binh,  
so_luong_ban, camera_truoc, ram, the_nho, dung_luong_pin, man_hinh_rong, don_gia  
FROM du_lieu_ban_hang_backup
```

	ID	ngay_ban	san_pham	khu_vuc	so_cuoc_goi	thoi_gian_trung_binh	so_luong_t
1	1	2018-09-19	BlackBerry KEY2	Bắc Ninh	398	354	2285
2	2	2018-09-20	BlackBerry KEY2	Bắc Ninh	206	406	2868
3	3	2018-09-21	BlackBerry KEY2	Bắc Ninh	224	366	3477
4	4	2018-09-22	BlackBerry KEY2	Bắc Ninh	350	320	3025
5	5	2018-09-23	BlackBerry KEY2	Bắc Ninh	304	337	3808
6	6	2018-09-24	BlackBerry KEY2	Bắc Ninh	324	384	1219
7	7	2018-09-25	BlackBerry KEY2	Bắc Ninh	232	315	3876

- 2NF: Để thỏa mãn 2NF, e tạo ra 2 bảng, 1 bảng gồm có ID (PRIMARY KEY), ngay\_ban, san\_pham\_ID, khu\_vuc, so\_cuoc\_goi, thoi\_gian\_trung\_binh, so\_luong\_ban và một bảng khác là thông tin sản phẩm gồm san\_pham\_ID (foreign key), san\_pham, camera\_truoc, ram, the\_nho, dung\_luong\_pin, man\_hinh\_rong, don\_gia
  - 1NF
  - Không có partial dependency:

```
CREATE TABLE san_pham_3NF (  
san_pham_ID INT IDENTITY(1,1) PRIMARY KEY,  
san_pham NVARCHAR(100),
```

```

camera_truoc INT,
ram INT,
the_nho INT,
dung_luong_pin INT,
man_hinh_rong FLOAT,
don_gia BIGINT,
)

```

```

INSERT INTO san_pham_3NF (san_pham, camera_truoc, ram, the_nho, dung_luong_pin,
man_hinh_rong, don_gia)

```

```

SELECT DISTINCT san_pham, camera_truoc, ram, the_nho, dung_luong_pin,
man_hinh_rong, don_gia FROM du_lieu_ban_hang_backup

```

(8 rows affected)

Total execution time: 00:00:00.166



	san_pham_ID	san_pham	camera_truoc	ram	the_nho	dung_luong_pin	man_hinh_rong
1	1	BlackBerry KEY2	8	4	128	3000	4.5
2	2	iPhone 11 64GB	12	4	128	3110	6.1
3	3	OPPO A93	16	8	256	4000	6.43
4	4	Realme 6 Pro	64	8	256	4300	6.6
5	5	Samsung Galaxy A21s	13	3	256	5000	6.5
6	6	Samsung Galaxy A50s	32	4	512	4000	6.4
7	7	Vivo V19 Neo	32	8	256	4500	6.44
8	8	Xiaomi Redmi Note 8	13	4	256	4000	6.3

3. 3NF: Để thỏa mãn 3NF, e sẽ tạo thêm 2 bảng nữa gồm ngay\_ban\_id, ngay\_ban và khu\_vuc\_id, khu\_vuc

1. 2NF

2. Không có functional dependency:

```

CREATE TABLE khu_vuc_3NF(
khu_vuc_ID INT IDENTITY(1,1) PRIMARY KEY,
khu_vuc NVARCHAR(100)
)

```

```

INSERT INTO khu_vuc_3NF (khu_vuc)

```

```

SELECT DISTINCT khu_vuc FROM du_lieu_ban_hang_backup

```

(4 rows affected)

Total execution time: 00:00:00.165



	khu_vuc_ID	khu_vuc
1	1	Bắc Ninh
2	2	Hải Phòng
3	3	Hà Nội
4	4	Nam Định

```
CREATE TABLE ngay_ban_3NF(  
ngay_ban_ID INT IDENTITY(1,1) PRIMARY KEY,  
ngay_ban DATE  
)
```

```
INSERT INTO ngay_ban_3NF(ngay_ban)  
SELECT DISTINCT ngay_ban FROM du_lieu_ban_hang_backup
```

(384 rows affected)

Total execution time: 00:00:00.169



	ngay_ban_ID	ngay_ban
1	1	2018-10-04
2	2	2018-09-11
3	3	2018-11-19
4	4	2018-10-27
5	5	2019-01-04
6	6	2018-12-12
7	7	2018-07-19
8	8	2018-01-22
9	9	2018-09-17
10	10	2018-06-03
11	11	2018-03-09

```
CREATE TABLE du_lieu_ban_hang_3NF (  
ID INT IDENTITY(1,1) ,  
ngay_ban_ID INT FOREIGN KEY REFERENCES ngay_ban_3NF (ngay_ban_ID),  
san_pham_ID INT FOREIGN KEY REFERENCES san_pham_3NF (san_pham_ID),  
khu_vuc_ID INT FOREIGN KEY REFERENCES khu_vuc_3NF (khu_vuc_ID),  
so_cuoc_goi INT,
```

```
thoi_gian_trung_binh INT,
so_luong_ban INT,
PRIMARY KEY(ngay_ban_ID, san_pham_ID, khu_vuc_ID)
)
```

```
INSERT INTO du_lieu_ban_hang_3NF (ngay_ban_ID, san_pham_ID, khu_vuc_ID,
so_cuoc_goi, thoi_gian_trung_binh, so_luong_ban)
SELECT ngay_ban_3NF.ngay_ban_ID, san_pham_3NF.san_pham_ID,
khu_vuc_3NF.khu_vuc_ID , so_cuoc_goi, thoi_gian_trung_binh, so_luong_ban
FROM ngay_ban_3NF, san_pham_3NF, khu_vuc_3NF, du_lieu_ban_hang_backup
WHERE du_lieu_ban_hang_backup.ngay_ban = ngay_ban_3NF.ngay_ban AND
du_lieu_ban_hang_backup.san_pham = san_pham_3NF.san_pham AND
du_lieu_ban_hang_backup.khu_vuc = khu_vuc_3NF.khu_vuc
```

(12285 rows affected)

Displaying Top 5000 rows.

Total execution time: 00:00:01.232



	ID	ngay_ban_ID	san_pham_ID	khu_vuc_ID	so_cuoc_goi	thoi_gian_trung_binh	so_luong_ban
1	1	301	1	1	398	354	2285
2	2	24	1	1	206	406	2868
3	3	268	1	1	224	366	3477
4	4	311	1	1	350	320	3025
5	5	73	1	1	304	337	3808
6	6	226	1	1	324	384	1219
7	7	369	1	1	232	315	3876
8	8	130	1	1	339	385	3366
9	9	272	1	1	125	272	1480
10	10	324	1	1	295	494	1972
11	11	187	1	1	123	221	2148
12	12	235	1	1	361	345	1625
13	13	55	1	1	245	346	1421
14	14	104	1	1	394	381	1982

### 3. Báo cáo T0+10

Tạo bảng SALES và PRODUCTS

```
CREATE TABLE PRODUCTS (  
san_pham NVARCHAR(100) PRIMARY KEY,  
camera_truoc INT,  
ram INT,  
the_nho INT,  
dung_luong_pin INT,  
man_hinh_rong FLOAT,  
don_gia BIGINT,  
)  
INSERT INTO PRODUCTS (san_pham, camera_truoc, ram, the_nho, dung_luong_pin,  
man_hinh_rong, don_gia)  
SELECT DISTINCT san_pham, camera_truoc, ram, the_nho, dung_luong_pin, man_hinh_rong,  
don_gia FROM du_lieu_ban_hang_backup
```

```
CREATE TABLE SALES (  
ngay_ban DATE,  
san_pham NVARCHAR(100),  
khu_vuc NVARCHAR(100),  
so_cuoc_goi INT,  
thoi_gian_trung_binh INT,  
so_luong_ban INT,  
PRIMARY KEY(ngay_ban, san_pham, khu_vuc),  
CONSTRAINT SALES_PRODUCTS FOREIGN KEY (san_pham) REFERENCES  
PRODUCTS(san_pham)  
)  
INSERT INTO SALES (ngay_ban, san_pham, khu_vuc, so_cuoc_goi, thoi_gian_trung_binh,  
so_luong_ban)  
SELECT ngay_ban, san_pham, khu_vuc, so_cuoc_goi, thoi_gian_trung_binh, so_luong_ban FROM  
du_lieu_ban_hang_backup
```

Để làm bài tập này, ta có thể dùng SQL Server Agent GUI hoặc T-sql command

**Cách 1:** SQL Server Agent GUI (ref: <https://database.guide/create-a-sql-server-agent-job-in-azure-data-studio/> và <https://stackoverflow.com/questions/5471080/how-can-i-schedule-a-job-to-run-a-sql-query-daily>):

Bởi em không biết SQL Server Agent nhìn như thế nào khi không bị khóa, nên e viết giải pháp dựa trên 2 nguồn phía trên.

- + Create New Job
- + Createa New Step
- + Setup command: ở đây e setup lệnh để tự xóa thông tin ở bảng SALES và đưa thông tin đó vào bảng HISTORY. Em vẫn để so sánh thời gian ở đây, vì giả sử hệ thống bị delay schedule, thì cách làm này vẫn đảm bảo chỉ di chuyển dữ liệu cũ của tháng trước, mà không di chuyển dữ liệu của tháng hiện tại.

```
INSERT INTO HISTORY  
SELECT D.*
```

```
FROM  
(DELETE SALES output deleted.*  
WHERE YEAR(ngay_ban) + MONTH(ngay_ban) < FORMAT(CAST(GETDATE() as date),  
'yyyyMM')) D
```

+ Setup Schedule: theo em đọc trên 2 posts thì có thể chọn monthly được, vào ngày đầu tháng và 1:00  
Trong trường hợp không có schedule thích hợp, em có thể setup sp\_add\_jobschedule như sau:  
@active\_start\_date và @active\_end\_date không được xác định

```
EXEC sp_add_jobschedule  
    @job_name = N'BackupSALES',  
    @name = N'Monthly_1_1AM',  
    @freq_type = 16,  
    @freq_INTERVAL = 1,  
    @freq_recurrence_factor = 1,  
    @active_start_time = 010000;
```

**Cách 2:** T-sql (ref: [https://docs.microsoft.com/en-us/previous-versions/sql/sql-server-2008-r2/ms181153\(v=sql.105\)?redirectedFROM=MSDN](https://docs.microsoft.com/en-us/previous-versions/sql/sql-server-2008-r2/ms181153(v=sql.105)?redirectedFROM=MSDN))

+ Chạy sp\_add\_job để tạo job

```
USE talentdb ;  
GO  
  
EXEC dbo.sp_add_job  
    @job_name = N'BackupSALES' ;  
GO
```

+ Chạy sp\_add\_jobstep

```
USE talentdb;  
GO  
EXEC sp_add_jobstep  
    @job_name = N'BackupSALES',  
    @step_name = N'SALES2HISTORY',  
    @subsystem = N'TSQL',  
    @command = N'  
INSERT INTO HISTORY  
SELECT D.*  
FROM  
(DELETE SALES output deleted.*  
WHERE YEAR(ngay_ban) + MONTH(ngay_ban) < FORMAT(CAST(GETDATE() as date), 'yyyyMM'))  
D',  
    @retry_attempts = 5,  
    @retry_INTERVAL = 5 ;  
GO
```

+ Chạy sp\_add\_schedule

```
USE talentdb ;  
GO  
  
EXEC dbo.sp_add_schedule  
    @schedule_name = N'OnceAMonth',
```



```
@freq_type = 16,  
@freq_INTERVAL = 1,  
@freq_recurrence_factor = 1,  
@active_start_time = 010000;
```

GO

+ Chạy sp\_attach\_schedule

```
USE talentdb ;  
GO
```

```
EXEC sp_attach_schedule  
    @job_name = N'BackupSALES',  
    @schedule_name = N'OnceAMonth' ;  
GO
```

+ Chạy sp\_add\_jobserver

```
USE msdb ;  
GO
```

```
EXEC dbo.sp_add_jobserver  
    @job_name = N'BackupSALES' ;  
GO
```

#### 4. Báo cáo T0 +14

Để viết thủ tục tự động, em sử dụng Procedure. Thông tin nên được trích xuất từ HISTORY, tuy nhiên do HISTORY không được cập nhật, em trích xuất trên SALES

1. Top 3 ngày có doanh số cao nhất

```
CREATE PROCEDURE baocao_ngay(@UserStart DATETIME, @UserEnd DATETIME)
AS BEGIN
```

```
SELECT TOP(3) ngay_ban, sum(so_luong_ban * don_gia)
FROM SALES
INNER JOIN PRODUCTS ON PRODUCTS.san_pham = SALES.san_pham
WHERE ngay_ban BETWEEN @UserStart AND @UserEnd
GROUP BY ngay_ban
```

```
END
```



	ngay_ban	(No column name)
1	2018-01-01	1315639560000
2	2018-01-02	1260714560000
3	2018-01-03	1235414560000

2. Top 3 sản phẩm doanh số cao nhất

```
CREATE PROCEDURE baocao_sp(@UserStart DATETIME, @UserEnd DATETIME)
AS BEGIN
```

```
SELECT TOP(3) SALES.san_pham, sum(so_luong_ban * don_gia)
FROM SALES
INNER JOIN PRODUCTS ON PRODUCTS.san_pham = SALES.san_pham
WHERE ngay_ban BETWEEN @UserStart AND @UserEnd
GROUP BY SALES.san_pham
```

```
END
```



	san_pham	(No column name)
1	Realme 6 Pro	23601851550000
2	Xiaomi Redmi Note 8	27090395130000
3	Vivo V19 Neo	57031290360000

3. Top 3 khu vực có doanh số kém nhất

```
CREATE PROCEDURE baocao_kv(@UserStart DATETIME, @UserEnd DATETIME)
AS BEGIN
```

```
SELECT TOP(3) khu_vuc, sum(so_luong_ban * don_gia)
FROM SALES
INNER JOIN PRODUCTS ON PRODUCTS.san_pham = SALES.san_pham
WHERE ngay_ban BETWEEN @UserStart AND @UserEnd
GROUP BY khu_vuc
ORDER BY sum(so_luong_ban) ASC
```

```
END
```



	khu_vuc	(No column name)
1	Hải Phòng	101239444520000
2	Bắc Ninh	102262407170000
3	Nam Định	102925978580000