FÖ6 – Fortsättning (T-)SQL

725G28: Databaser och datamodellering

Jonathan Crusoe

Informatik, IEI, Linköpings Universitet

LINKÖPINGS UNIVERSITET

Last time on SQL…

```
INSERT INTO staff (staff_num, first_name, phone_num)
VALUES (1, 'Sotis', '0163474568'),
       (2, 'Boris The Bader', '070181822'),
       (3, 'Louise', '010223344');

SELECT *
  FROM staff
 WHERE phone_num LIKE '01%';

UPDATE staff
   SET first_name = 'Kalle Anka'
 WHERE phone_num LIKE '%0%'
    OR first_name LIKE 'Sot%'

DELETE FROM staff
 WHERE first_name = first_name
```

Vad är problemet med UPDATE och DELETE?

Agenda

Ämne	Kapitel
Logical Processing Order	-
Nästlade frågor & Alias	7
Join	8
Standard & Aggregations Funktioner	8
Kalkylering	-
Vyer	7
Användar Definerade Funktioner	14
Lagrade Procedurer	14
Triggers	15
Praktiska Exempel	-

https://docs.microsoft.com/sv-se/sql/t-sql/language-elements/language-elements-transact-sql

Logical Processing Order of SELECT-statments

Standard SELECT-statment

```
SELECT DISTINCT TOP N C
  FROM T1
       JOIN T2
       ON B1
 WHERE B2
 GROUP BY G
HAVING B3
 ORDER BY C
```

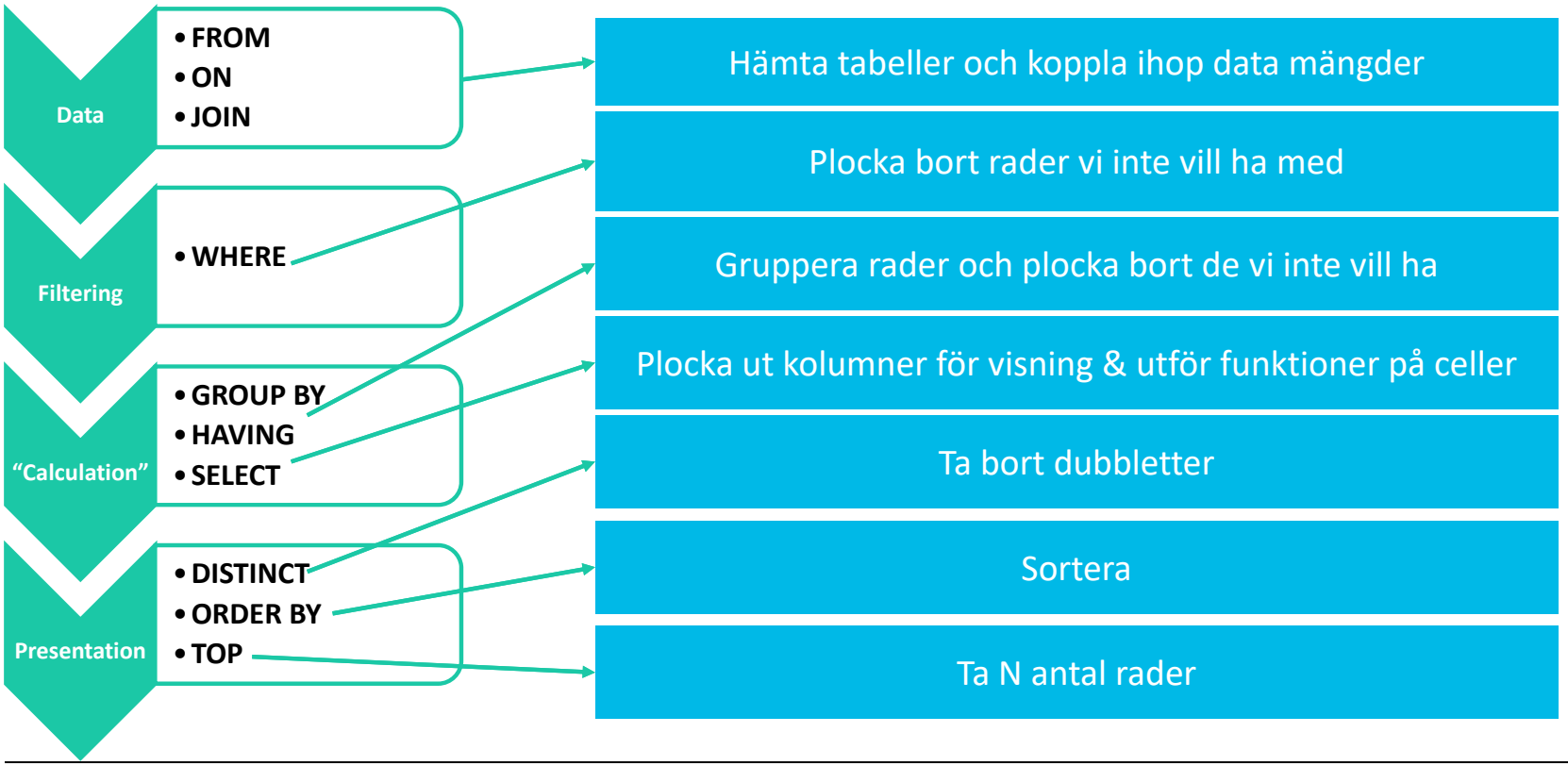
Hur vi läser

Hur datorn läser

ORD.	Nyckelord	Värde
1	FROM	T1
2	ON	B1
3	JOIN	T2
4	WHERE	B2
5	GROUP BY	G
6	HAVING	B3
7	SELECT	C
8	DISTINCT	-
9	ORDER BY	C
10	TOP	N

* Varför denna skillnad?

Logical Processing Order of SELECT-statments



https://docs.microsoft.com/en-us/sql/t-sql/queries/select-transact-sql

music					
music_num	song_title	artist	album	review_rank	price
1	Trump Bing Bong Remixes	Bosco	YouTube	8 of 10	10$
2	I'm an Albatraoz	AronChupa	I'm an Albatraoz	1 of 5	 $
3	Fanten	Jonatan Ersarp	Unknown	3 of 7	1$
…	…	…	…	…	…

sales_item			
sales_num	music_num	movie_num	quantity
1	1	null	2
1	null	2	10
2	null	3	1
…	…	…	…

movie			
movie_num	title	review_rank	price
1	I det hetaste laget	5 of 5	10$
2	Frozen	3 of 5	5$
3	Star Trek: Discovery	2 of 6	1$
…	…	…	…

sales				
sales_num	sales_date	discount	staff_num	customer_num
1	Maj-17	10	1	2
2	Jun-20	0	4	5
3	Apr-12	100	1	18
…	…	…	…	

Näslad Fråga & Alias

```
SELECT discount, (
       SELECT first_name
         FROM staff
        WHERE staff_num = S.staff_num
       ) AS STAFF
  FROM sales AS S;
```

Result	
discount	**STAFF**
100	Sotis
10	Boris The Badger
0	Louise

Kan användas i SELECT, FROM, WHERE, och GROUP BY

>> UNDVIK GÄRNA <<
De är dyra och svår lästa…

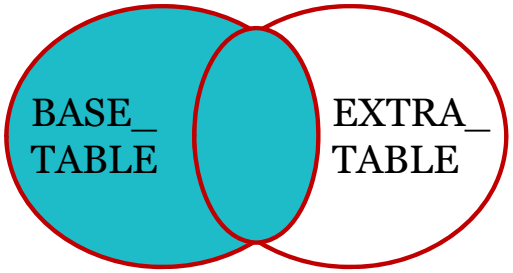
- Nested Queries (Subqueries)
  - Måste komma ihåg att binda ihop!
  - Subqueries är ”temporära” tabeller vi plockar ihop = DYRT!
- Alias (AS)
  - AS kan användas för att döpa om kolumner i resultat (T.ex. STAFF)
  - AS kan användas för att skapa kortare namn (T.ex. S) i queries

Data → Filtering → "Calculation" → Presentation

Join

```
SELECT * FROM [BASE_TABLE] [SUBSET] JOIN [EXTRA_TABLE] ON [WHEN?];
```

- Binder ihop datamängder till en ”temporär/virtuell” tabell
- En grov uthuggning av den datan vi vill arbeta med
- Vi väljer bas och extra
- Vi väljer vilka [subset] vi vill arbeta med
- Vi bestämmer hur de kopplar samman [WHEN?]

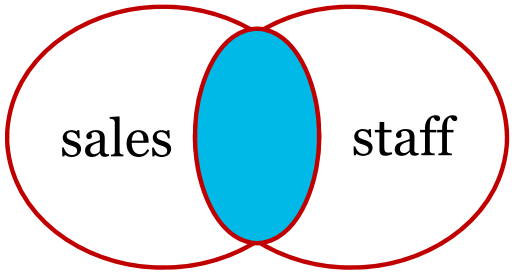


Inre Join – ”Vanlig Join” – INNER JOIN

- Binder snyggt ihop tabeller

”AS” används för att skapa alias för tabeller och kolumner
I SELECT kan de döpa om kolumnen för presentation

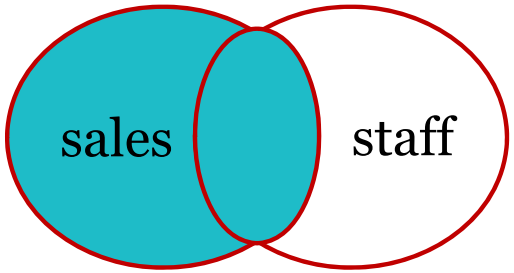
```
SELECT SA.discount, ST.first_name AS STAFF
  FROM sales AS SA
       INNER JOIN staff AS ST
       ON SA.staff_num = ST.staff_num;
```



Result	
discount	**STAFF**
100	Sotis
10	Boris The Badger
0	Louise

Outer Join

```
SELECT SA.discount, ST.first_name AS STAFF
  FROM sales AS SA
       LEFT OUTER JOIN staff AS ST
       ON SA.staff_num = ST.staff_num;
```

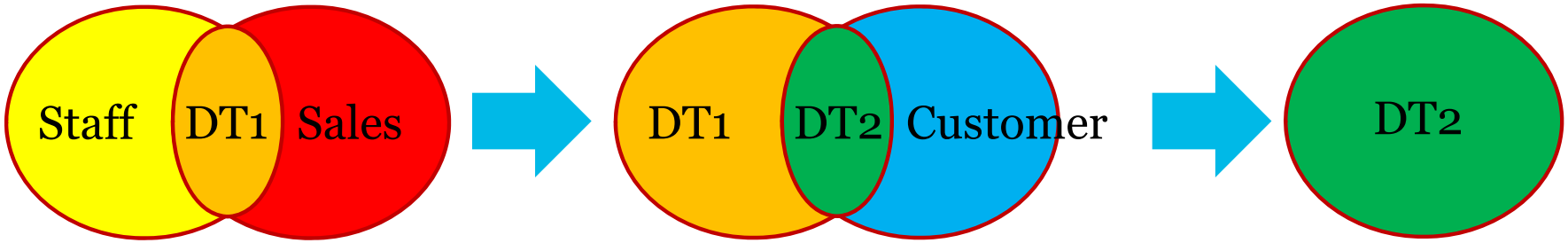


Om en staff inte finns för en discount blir resultatet null
(Right kan ändras till Left)

Result	
discount	**STAFF**
100	Sotis
10	Boris The Badger
0	Louise
23	null

Multiple Join

```
SELECT staff.first_name, sales_date, customer.first_name
  FROM staff
       INNER JOIN sales
       ON staff.staff_num = sales.staff_num
       INNER JOIN customer
       ON sales.customer_num = customer.customer_num
```



WHERE vs. JOIN

```
SELECT SA.discount, ST.first_name AS STAFF
  FROM sales AS SA
       INNER JOIN staff AS ST
       ON SA.staff_num = ST.staff_num;

SELECT SA.discount, ST.first_name AS STAFF
  FROM sales AS SA, staff AS ST
 WHERE SA.staff_num = ST.staff_num
```

- JOIN (Explicit join)
  - läsbar och lättare att ändra arbetsdatamängder
- WHERE (Implicit join)
  - kompaktare och filtrerar allt på en plats
- BÅDA
  - ger samma resultat

WHERE (Från tidigare FÖ)

- Comparison Operators

  = < > <= >= <> !=

- Arithmetic Operators

  + - * /

- Functions

  Count Sum Avg Max Min

- Nested statements

  ( ) e.g. NOT(((cond1) AND (cond2)) OR cond3)

- Boolean Operators

  AND OR NOT

- New variables

  AS eg. 2/3 AS twothirds

WHERE (Från tidigare FÖ)

Name	Description
[NOT] BETWEEN…AND…	Check whether a value is within or not within a range of values
LIKE	Simple pattern matching
%	Wildcard in comparisons that match
^	Wildcard for comparisons that do not match
-	Wildcard for comparisons that match one character
[a-z] or [0-9]	The content within the square brackets gives the range

(https://docs.microsoft.com/en-us/sql/t-sql/language-elements/string-operators-transact-sql)

Data | Filtering | "Calculation" | Presentation

WHERE (Kapitel 8)

Name	Description
[NOT] IN	Finns värdet I tabellen
[NOT] EXISTS	Finns det något värde I tabellen
Värde *jämförelse* ANY/SOME/ALL table	Jämför ett värde mot flera värden
ANY	Jämför ett värde mot flera för att hitta någon som stämmer
SOME	Samma som ANY
ALL	Jämför ett värde mot alla värden i en annan tabell

(https://docs.microsoft.com/en-us/sql/t-sql/language-elements/string-operators-transact-sql)

WHERE

```
SELECT staff.first_name, sales_date, customer.first_name
  FROM staff
       INNER JOIN sales
       ON staff.staff_num = sales.staff_num
       INNER JOIN customer
       ON sales.customer_num = customer.customer_num
 WHERE sales.sales_date BETWEEN '2011-01-01' AND '2017-01-01'
   AND sales.discount > 0;
```

* Finns det något ni undrar över kring WHERE?

(https://docs.microsoft.com/en-us/sql/t-sql/language-elements/string-operators-transact-sql)

Standard & Aggregations Funktioner

- JOIN - plockar ut mängden vi vill räkna med
  - T.ex. Staff & Sales ihopkopplat genom staff_num
- WHERE - plockar ut vad vi vill räkna med
  - T.ex. Alla anställda inom års perioden 2011 till 2017
- GROUP BY – Över vilka kolumner vill vi slå ihop rader
  - T.ex. Anställdas namn, Köpares bostads adress, Produkt kostnad…

- Men vad händer om vi har flera olika värden bland raderna som slås ihop?

Standard & Aggregations Funktioner

- Aggregations Funktioner (Mängdoperationer)
  - Slår ihop flera rader till en genom uträkning
  - Utan `GROUP BY` slår vi ihop alla rader till en
- `MIN` – Minsta värdet i en grupp/kolumn
- `MAX`– Största värdet i en grupp/kolumn
- `AVG` – Medelvärdet i en grupp/kolumn
- `SUM` – Summan av en grupp/kolumn
- `COUNT` – Antal värden i en grupp/kolumn
- EVERY, SOME, och ANY (i boken) - Funkar inte på vår server

Standard & Aggregations Funktioner

```
SELECT SUM(sales.discount) AS TOT_DISC
  FROM sales;
```

Result
TOT_DISC
110

- MIN – Minsta rabatt på en försäljning bland alla försäljningar
- MAX– Största rabatt på en försäljning bland alla försäljningar
- AVG – Medelrabatt för alla försäljningar
- SUM – Rabatt summan för alla försäljningar
- COUNT – Totala antalet av alla försäljningar

Standard & Aggregations Funktioner

- Kombinerar vi med `GROUP BY` kan vi dela upp helheten
  - T.ex. Rabatt per månad

```
  SELECT MONTH(sales.sales_date) AS Month,
         SUM(sales.discount) AS TOT_DISC
    FROM sales
GROUP BY MONTH(sales.sales_date);
```

Result	
Month	**TOT_DISC**
10	100
11	10

* Vad är problemet med frågan?

Standard & Aggregations Funktioner

(https://docs.microsoft.com/en-us/sql/t-sql/functions/functions)

Standard Funktion	Resultat
ABS(n)	Omvandlar negativa tal till positiva
CEILING(n)	Avrundar uppåt t.ex. 1.05 → 2
FLOOR(n)	Avrundar nedåt t.ex. 1.05 → 1
RAND(n)	Slumpmässigt värde
REPLACE(str, pattern, replacement)	Bytt ut delar i en sträng
LEN(str)	En strängs längd
LOWER(str)	Sträng till små bokstäver
SUBSTRING(str, start, length)	Plockar ut en substräng
DAY(date)	Retunerar dagnummret som int
MONTH(date)	Retunerar månadsnummret som int

Standard & Aggregations Funktioner

```
SELECT CONCAT('BOOK', 'DOG', 'CAT');
SELECT 'BOOK' + 'DOG' + 'CAT';
```

→ BOOKDOGCAT

```
SELECT COALESCE(null, null, 'KAKA', 'HUND');
SELECT COALESCE(null, 'KATT', 'KAKA', 'HUND');
```

→ KAKA

→ KATT

Returnerar första värdet som inte är null!

Kalkylering

Väldigt likt java fast med lite andra namn på funktionerna

- Vi kan utföra matematiska uträckningar
  - T.ex. +, -, *, /
- Fungerar i SELECT, ON, WHERE, GROUP BY, HAVING

```
SELECT quanitity * cost AS price FROM ...

... WHERE LEN(first_name) * 4 < LEN(last_name) * 4 ...

SELECT COUNT(*) AS 'Amount', (LEN(first_name) * 4 - 2) AS 'Length'
  FROM staff
 GROUP BY LEN(first_name) * 4 - 2;
```

Stökigt att utföra kalkyler i group by…

Presentation - Tidigare Föreläsning

- DISTNICT
  - Unika värden
- ORDER BY
  - Sorterar värdena efter kolumn
    - DESC – DESCENDING
    - ASC – ASCENDIN
- TOP (NY!)
  - Tar ut ett antal rader

```
SELECT DISTINCT TOP 2 *
  FROM staff
 ORDER BY first_name DESC
```

Vyer → `CREATE VIEW [NAME] AS [SELECT QUARY]...`

- Är en tabell som inte lagras i databasen
- Räknas ut varje gång man kallar på den

- Kan användas för att…
  - slippa att göra frågor med upprepade nästlade frågor
  - slippa skriva frågor för standard frågor upprepade gånger

- I grunden är det en SELECT som gjorts till en virtuell tabell

Skapa vy

```
CREATE VIEW trades AS
     SELECT SA.sales_num,
            (ST.first_name) AS seller,
            (CO.first_name) AS buyer
       FROM sales AS SA
            INNER JOIN staff AS ST
            ON SA.staff_num = ST.staff_num
            INNER JOIN customer AS CO
            ON SA.customer_num = CO.customer_num
```

```
SELECT * FROM trades;
```

Result		
sales_num	**seller**	**buyer**
1	Sotis	Roland
2	Boris The Badger	Dimitri
3	Louise	Phestive
…	…	…

Vi har gjort om en SELECT till en ”virtuell” tabell

Skapa vy

```
CREATE VIEW sales_total AS
     SELECT SA.sales_num,
            SUM(SI.quantity) AS total,
            COUNT(SI.sales_num) AS amount
       FROM sales AS SA
            INNER JOIN sales_item AS SI
            ON SA.sales_num = SI.sales_num
   GROUP BY SA.sales_num;
```

```
SELECT * FROM sales_total;
```

Result		
sales_num	**total**	**amount**
1	25	2
2	15	2
3	9	2
…	…	…

SELECT & Views

```
SELECT T.sales_num, T.buyer, T.seller, S.total
  FROM trades AS T
       INNER JOIN sales_total AS S
       ON T.sales_num = S.sales_num;
```

Viewes är ”virtuella” tabeller och kan behandlas där efter

Result			
sales_num	**buyer**	**seller**	**Total**
1	Roland	Sotis	25
2	Dimitri	Boris The Bader	15
3	Phestive	Louise	9
…	…	…	…

Scalar Function & Stored Procedure

- Utökning av SQL
  - Funktion, Procedur och Trigger använder samma kodspråk (Skiljer från boken)
  - Chunky att arbeta med; Viktigt att test köra mycket
- Klarar…
  - hierarkiska steg uträckningar - *Transitivt hölje*
  - iterationer - *Transitivt hölje*
  - jämföra rader och skriva uträckningar på en tredje
- Funktionell programmering (likt Java)
  - Tabell ≈ Klass
  - Rad ≈ Objekt
  - Saknar beteende och ”collections”

DML – SELECT, INSERT, UPDATE, DELETE

Scalar Function & Stored Procedure

- Stored Procedures
  - Kan inte användas i DML satser
  - Kan utföra DML satser
  - Kan ha flera returvärden
- Scalar Function ≈ Användar Definerade Funktioner
  - Använder samma språk som Stored Procedures
  - Kan användas i DML satser
  - Kan inte utföra DML satser
  - Kan bara ha ett returvärde
- ”Metodhuvud” och ”Metodslut” skrivs lite olika…

Scalar Function (≈ Användar Definierade Funktioner)

```
CREATE FUNCTION function_name (@var1 VARCHAR(10), @var2 INT)
       RETURNS VARCHAR(10)
    AS
       BEGIN
          -- HÄR SKRIVS KODEN! :)
          RETURN @some_return_value;
       END;
```

Anropa en funktion: `SELECT function_name('TST', 2);`

Stored Procedure

```
CREATE PROCEDURE function_name
       @var1 VARCHAR(10),
       @var2 INT,
       @some_return_value VARCHAR(10) OUTPUT
    AS
       -- HÄR SKRIVS KODEN! :)
    RETURN;
```

DEFINITION

```
DECLARE @output VARCHAR(10);

EXEC function_name
     “TST",
     2,
     @output OUTPUT;

SELECT @output AS 'VALUE';
```

ANROP

Triggers

- Aktiva Databaser
  - Databashanteraren kan utföra egna handlingar
- Aktiva regler (Triggers) består av tre delar
  - Händelse
  - Villkor
  - Åtgärd/Handling
- Aktiveras per tabell i T-SQL

Triggers

- Exekveras automatiskt
  - Oftast på INSERT, UPDATE, DELETE
  - Finns för CREATE, ALTER, DROP

```
CREATE TRIGGER [trigger_name]
            ON [table_name] -- What object
           FOR / AFTER / INSTEAD OF -- When
               INSERT / UPDATE / DELETE -- What action (EVENT)
            AS
         BEGIN
               -- IF-statement if needed (CONDITION).
               -- SQL code goes here (How; ACTION).
               -- Just like a Stored Procedure
           END;
```

Triggers

```
CREATE TRIGGER [sales_cleaner]
            ON [sales]
           FOR DELETE
            AS BEGIN
               DELETE FROM [sales_item]
                WHERE [sales_item].sales_num IN (
                      SELECT deleted.sales_num FROM deleted
                      );
   END;
```

- Innan vi tar bort en sale tar vi bort alla relaterade sale_items
  - Kan lättare göras med CASCADE

Triggers

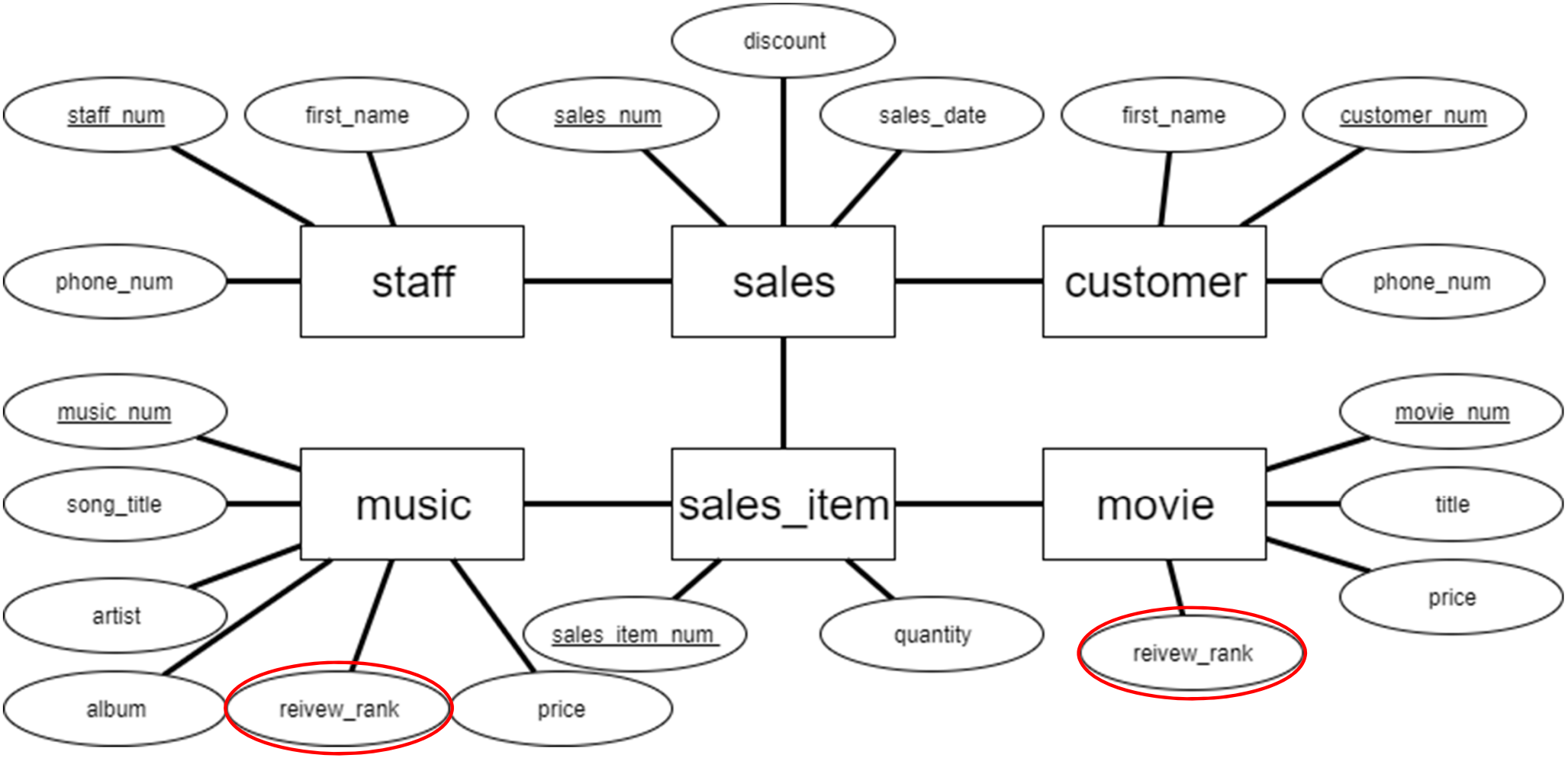
Nivå som förväntas för VG

```
CREATE TRIGGER [staff_cleaner]
            ON [staff]
         AFTER DELETE
            AS BEGIN
               INSERT INTO [former_staff] (staff_num, first_name, phone_num)
                    SELECT D.staff_num, D.first_name, D.phone_num
                      FROM deleted AS D
   END;
```

- Varje anställd som tas bort flyttas över till former_staff tabellen
  - Måste göras med TRIGGER

Praktiska Exempel

Notera: Har komplicerat situationen för att vissa funktionalitet. Ert projekt kommer inte vara så här svårt.


discount
staff_num
first_name
sales_num
sales_date
first_name
customer_num
phone_num
staff
sales
customer
phone_num
music_num
movie_num
song_title
music
sales_item
movie
title
artist
price
sales_item_num
quantity
reivew_rank
album
reivew_rank
price

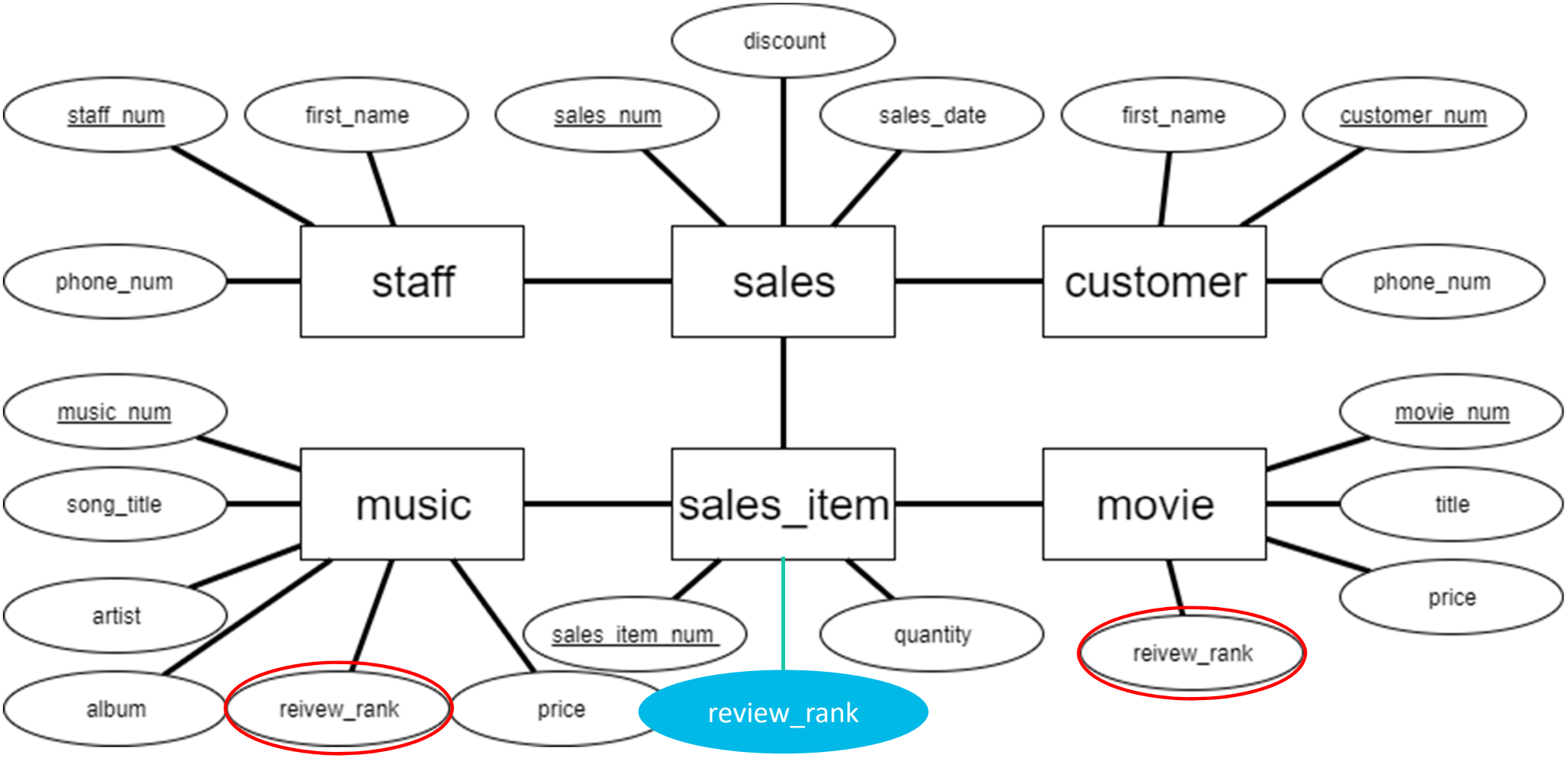
Mejl från chefen…

1. Lägga till en ny kolumn på sales_item, med constraint.
2. Flytta värden från music/movie till sales_item
3. Ta bort gamla kolumener

Hej IT-support,

Efter en kundundersökning har vi bestämt att flytta review systemet från expertomdöme till kundomdöme. Istället för att varje produkt har en review_rank, vill vi att varje kund ska kunna lämna ett review_rank vid köp. De ska kunna ranka en produkt på en 5-skala.

M.v.h. Chefen


discount
staff_num
first_name
sales_num
sales_date
first_name
customer_num
phone_num
staff
sales
customer
phone_num
music_num
movie_num
song_title
music
sales_item
movie
title
artist
price
sales_item_num
quantity
reivew_rank
album
reivew_rank
price
review_rank

music					
music_num	song_title	artist	album	review_rank	price
1	Trump Bing Bong Remixes	Bosco	YouTube	8 of 10	10$
2	I'm an Albatraoz	AronChupa	I'm an Albatraoz	1 of 5	 $
3	Fanten	Jonatan Ersarp	Unknown	3 of 7	1$
…	…	…	…	…	…

sales_item			
sales_num	music_num	movie_num	quantity
1	1	null	2
1	null	2	10
2	null	3	1
…	…	…	…

movie			
movie_num	title	review_rank	price
1	I det hetaste laget	5 of 5	10$
2	Frozen	3 of 5	5$
3	Star Trek: Discovery	2 of 6	1$
…	…	…	…

sales				
sales_num	sales_date	discount	staff_num	customer_num
1	Maj-17	10	1	2
2	Jun-20	0	4	5
3	Apr-12	100	1	18
…	…	…	…	

* Ser ni något problem?

1. Lägga till en ny kolumn på sales_item, med constraint.

```
ALTER TABLE sales_item ADD
          review_rank INT,
                      CONSTRAINT review_rank
                      CHECK(0 <= review_rank AND review_rank <= 5)
```

Värden under 0 och över 5 kan inte läggas in i kolumnen
(Går att lägga in när man skapar tabellen)

2. Flytta värden från music/movie till sales_item

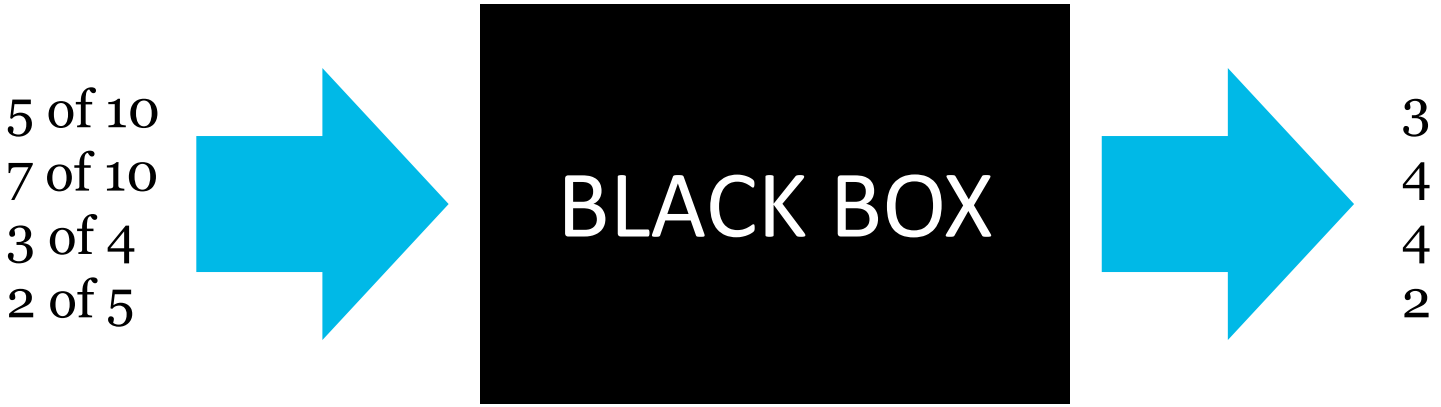
movie	
movie_num	review_rank
1	5 of 5
2	3 of 5
3	2 of 6
…	…

music	
music_num	review_rank
1	8 of 10
2	1 of 5
3	3 of 7
…	…

sales_item			
sales_num	music_num	movie_num	review_rank
1	1	null	null
1	null	2	null
2	null	3	null
…	…	…	null

1. Reviews är inte mellan 1 till 5
2. Värdet finns i två olika tabeller och ska in i en
3. Värdena ska gå från String till Int

1. Reviews är inte mellan 1 till 5

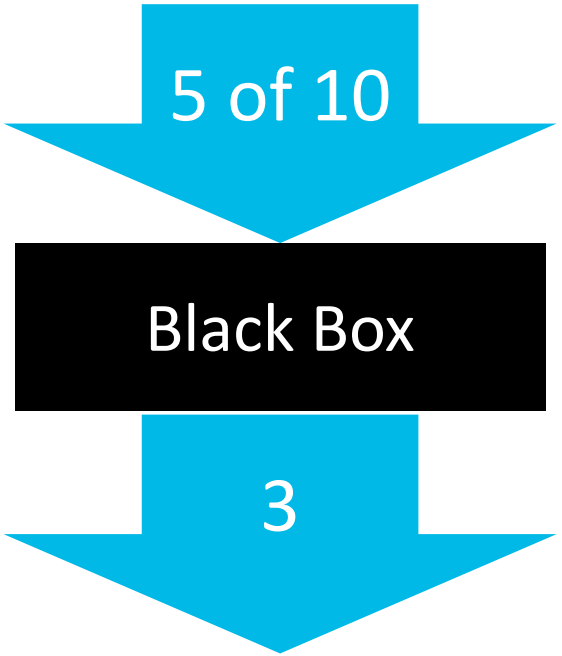


Ska den vara funktion eller procedur?

Function (Reviews är inte mellan 1 till 5)

```
SELECT [transform_review_rating]('5 of 10');
```

Test köra funktionen



Function (Reviews är inte mellan 1 till 5)

```
CREATE FUNCTION [transform_review_rating] (@review_rank VARCHAR(10))
       RETURNS VARCHAR(10)
       AS BEGIN
          DECLARE @new_review_rank VARCHAR(10);

          -- HÄR KOMMER KODEN! :)

          RETURN @new_review_rank;
END;
```

Grunden för funktionen

```
public String transform_review_rating (String review_rank) {

      // HÄR KOMMER KODEN! ☺

      return new_review_rank;
}
```

Samma fast i JAVA

Function (Reviews är inte mellan 1 till 5)

```
SET @review_rank = REPLACE(@review_rank, ' of ', '#');

DECLARE @DataSource TABLE (
      [id] INT IDENTITY(1,1),
      [value] NVARCHAR(4)
);

INSERT INTO @DataSource ([value])
       (SELECT [value] FROM STRING_SPLIT(@review_rank, '#'));
```

Vi använder en tabel som en array

5 of 10 → 5#10

[id]	[value]

5#10 →

[id]	[value]
1	5
2	10

IDENTITY (START_VALUE, INCREMENT)

Function (Reviews är inte mellan 1 till 5)

```
DECLARE @var1 FLOAT;
DECLARE @var2 FLOAT;

SET @var1 = (SELECT (CONVERT(FLOAT, [value]))
                FROM @DataSource WHERE [id] = 1);
SET @var2 = (SELECT (CONVERT(FLOAT, [value]))
                FROM @DataSource WHERE [id] = 2);
```

[id]	[value]
1	5
2	10

var1 = 5

var2 = 10

I Java...

```
float var1 = Float.valueOf(dataSource[0]);
float var2 = Float.valueOf(dataSource[1]);
```

Function (Reviews är inte mellan 1 till 5)

```
SET @var2 = @var2 / 5.0;
SET @var1 = @var1 / @var2;
```

float var2 = 10 / 5.0 → 2
float var1 = 5 / 2 → 2.5

```
SET @new_review_rank = (CONVERT(VARCHAR, CEILING(@var1)));

RETURN @new_review_rank;
```

Avrundar 2.5 till 3 med CEILING
Konventerar sen från Float till Varchar.

Varför avrundar jag uppåt?

Function (Reviews är inte mellan 1 till 5)

```
UPDATE music
   SET review_rank = [transform_review_rating](review_rank);
UPDATE movie
   SET review_rank = [transform_review_rating](review_rank);
```

2. Flytta värden från music/movie till sales_item

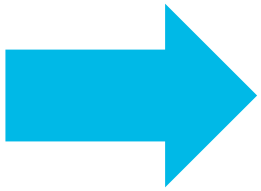
```
SELECT SI.sales_num,
       MO.review_rank AS movie_rank,
       MU.review_rank AS music_rank
  FROM sales_item AS SI
       LEFT OUTER JOIN movie AS MO
       ON SI.movie_num = MO.movie_num
       LEFT OUTER JOIN music AS MU
       ON SI.music_num = MU.music_num;
```

Results		
sales_num	**music_rank**	**movie_rank**
1	5	null
1	null	1
2	2	null
2	null	4
3	3	null
3	null	3
…	…	…

2. Flytta värden från music/movie till sales_item

Byte bara huvudet på satsen!

```
SELECT SI.sales_num,
       MO.review_rank AS movie_rank,
       MU.review_rank AS music_rank
```



```
UPDATE SI
   SET review_rank = COALESCE(MO.review_rank,
MU.review_rank, 0)
```

Retunerar första som inte är null

```
UPDATE SI
   SET review_rank = COALESCE(MO.review_rank, MU.review_rank, 0)
  FROM sales_item AS SI
       LEFT OUTER JOIN movie AS MO
       ON SI.movie_num = MO.movie_num
       LEFT OUTER JOIN music AS MU
       ON SI.music_num = MU.music_num;
```

3. Ta bort gamla värden…

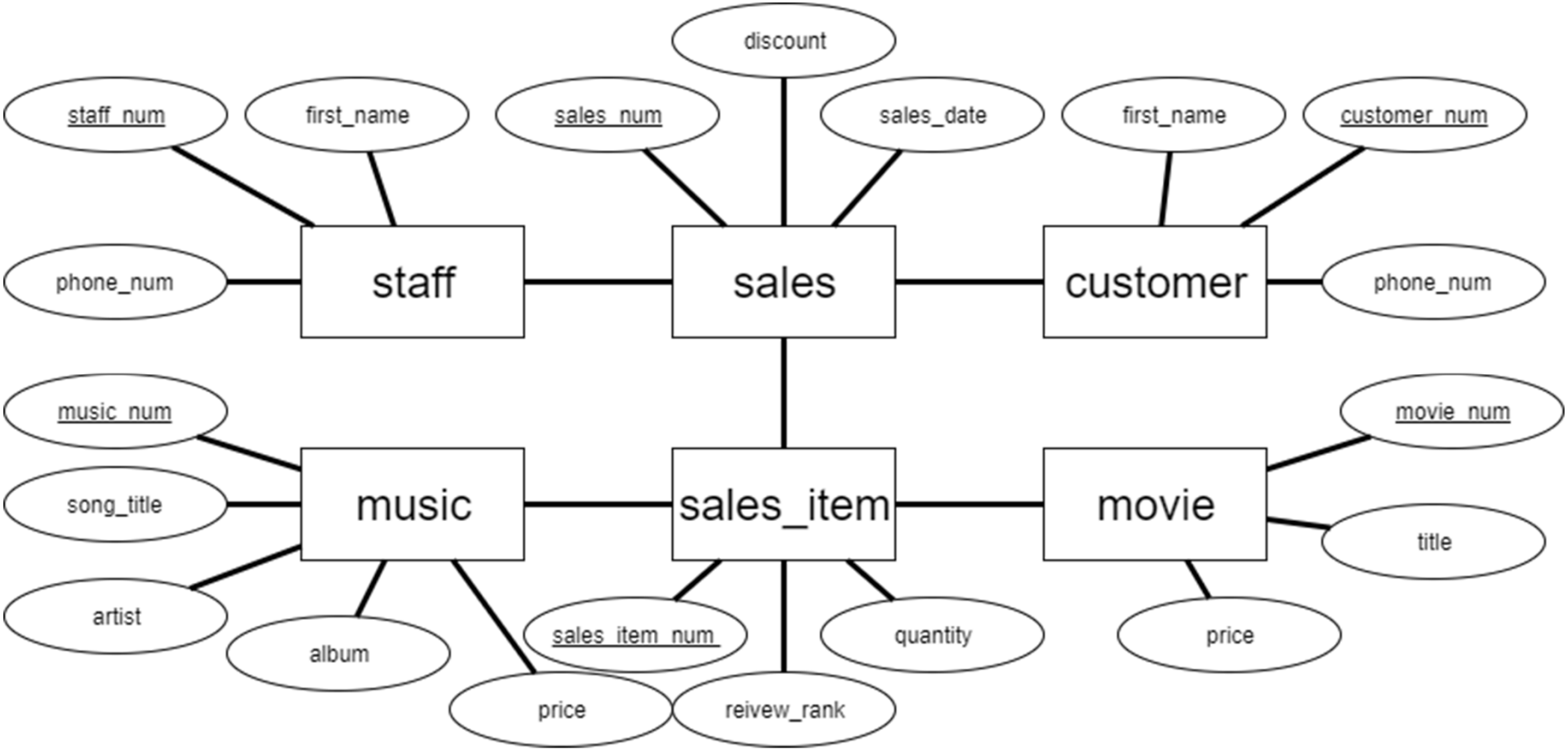
```
ALTER TABLE music
DROP COLUMN review_rank;

ALTER TABLE movie
DROP COLUMN review_rank;
```

Mejl från kundsupporten…

Hej IT-support,

Vi har fått flera klagomål på att man inte kan se granskningsrankningen på vår hemsida. Skulle ni kunna fixa det?

M.v.h. Simon på Kundsupporten



* Måste vi lösa problemet med en eller två viewes? Varför?

Två views…

```
CREATE VIEW movie_sales AS
     SELECT M.movie_num,
            M.title,
            SUM(SI.quantity) AS 'Total Sold',
            AVG(SI.review_rank) AS 'Avg Rank'
       FROM movie AS M
            LEFT OUTER JOIN sales_item AS SI
            ON M.movie_num = SI.movie_num
      GROUP BY M.movie_num, M.title;
```

```
CREATE VIEW music_sales AS
     SELECT M.music_num,
            M.song_title AS 'Song Title',
           SUM(SI.quantity) AS 'Total Sold',
           AVG(SI.review_rank) AS 'Avg Rank'
           FROM music AS M
           LEFT OUTER JOIN sales_item AS SI
           ON M.music_num = SI.movie_num
     GROUP BY M.music_num, M.song_title;
```

* Vad blir resultatet? Vad är logiken bakom vyerna?

Ett till mejl från chefen…!

Hej IT-enheten,

Efter ett möte med marknadsavdelningen vill vi ha statistik över produkt köp. Vi vill ha detta månadsvis. Tror du att det går att fixa en rapport på top tio för en månad?

M.v.h. Chefen

1. En ny tabell för statistik
2. En ökar metod för att öka statistik
3. En ny trigger för när någon köper

Statistik Tabellen…

DEFAULT låter mig sätta automatiska värden vid en INSERT.

```
CREATE TABLE sales_statistics (
   sale_year  DATE DEFAULT(CONVERT(VARCHAR, YEAR(GETDATE()) + '-01-01'))          NOT NULL,
   sale_month DATE DEFAULT(CONVERT(VARCHAR, '0000-' + MONTH(GETDATE()) + '-01'))  NOT NULL,
   category   VARCHAR(10)                                                         NOT NULL,
   item_num   INT                                                                 NOT NULL,
   amount     INT  DEFAULT(0)
);
```

* Vad hade varit en bättre lösning i det här fallet?

Procedur för att öka statistiken…

```
CREATE PROCEDURE [statistics_increase]
  @cat VARCHAR(10),
 @item INT
AS
DECLARE @year  DATE = CONVERT(VARCHAR, YEAR(GETDATE())) + '-01-01'
DECLARE @month DATE = '0001-' + CONVERT(VARCHAR, MONTH(GETDATE())) + '-01'

      -- IF STATEMENT (Add new row or update old)

 RETURN;
```

← Parametrar

Standard värden →

```
-- IF STATEMENT (Add new row or update old)
IF 0 = (SELECT COUNT(*)
          FROM sales_statistics AS stat
         WHERE stat.sale_year = @year
           AND stat.sale_month = @month
           AND stat.category = @cat
           AND stat.item_num = @item)
   INSERT INTO sales_statistics (category, item_num, amount)
   VALUES (@cat, @item, 1);
ELSE
   UPDATE sales_statistics
      SET amount += 1
    WHERE sale_year = @year
      AND sale_month = @month
      AND category = @cat
      AND item_num = @item
```

Finns det någon rad? Lägg till om den inte finns

Ignorerar vi WHERE-satsen så är det inte mycket kod!

Ökar om det finns

Början av triggern…

```
CREATE TRIGGER [statistics_adder]
            ON [sales_item]
         AFTER INSERT
            AS BEGIN
               DECLARE cursos CURSOR FOR
                           SELECT sales_num, music_num, movie_num
                             FROM inserted

               DECLARE @snum int, @music int, @movie int

               OPEN cursos;

               FETCH NEXT FROM cursos INTO @snum , @music , @movie
```

Trigger kör efter att någon lagt till en mängd rader i sales_item

← Vad vi itererar över

Sätter upp för att iterera genom alla tillagda rader

← Nuvarande Rad Hållare

← ”Väcker” iteratorn. Dags att jobba!

← Hämtar Första Raden

Iterationen i triggern…

```
WHILE @@FETCH_STATUS = 0
      BEGIN

      IF @music IS NOT NULL
         EXEC [statistics_increase] 'music', @music;
      IF @movie IS NOT NULL
         EXEC [statistics_increase] 'movie', @movie;

      FETCH NEXT FROM cursos INTO @snum , @music , @movie
END;
```

← Finns det kvar rader att hämta?

Itererar igenom varje ny rad. Kollar om de är musik eller film genom att de inte är null.

← Hämtar Nästa Rad

Slutet av triggern…

```
    CLOSE cursos;
    DEALLOCATE cursos;
END;
```

← ”Söver” iteratorn. Dags att sova!

← Dealokerar minnet som används…

Städar upp efter iteratorn ☺

Lite Test Körning!

```
INSERT INTO sales_item (sales_num, quantity, music_num, movie_num)
VALUES (3, 13, null, 1),
       (3, 12, 2, null),
       (2,  8, null, 3),
       (2,  7, 1, null),
       (1,  4, null, 2),
       (1, 5, 3, null),
       (3, 13, null, 1),
       (3, 12, 2, null),
       (2,  8, null, 3),
       (2,  7, 1, null),
       (1,  4, null, 2),
       (1, 5, 3, null);
```

```
SELECT * FROM sales_statistics;
```

	sale_year	sale_month	category	item_num	amount
1	2017-01-01	0001-11-01	music	3	2
2	2017-01-01	0001-11-01	movie	2	2
3	2017-01-01	0001-11-01	music	1	2
4	2017-01-01	0001-11-01	movie	3	2
5	2017-01-01	0001-11-01	music	2	2
6	2017-01-01	0001-11-01	movie	1	2

Jonathan Crusoe

Tack för att ni deltog! ☺

www.liu.se

LINKÖPINGS UNIVERSITET