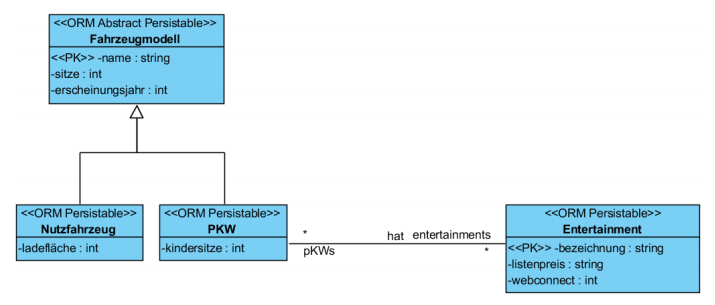
Ausarbeitung Übung 03

# Abbildung Generalisierung



Die Generalisierung ist *complete*, da das Fahrzeugmodell als abstrakte Klasse modelliert ist. Sie ist außerdem *disjoint*, da ein Modell laut Angabe in beide Sparten PKW **und** Nutzfahrzeug fallen kann. Beim Einrelationenmodell wäre die Beziehung **hat** polymorph, was gegen die Modellierung spricht. Durch die Vollständigkeit der Generalisierung bietet sich die volle Redundanz eher schlecht an. Aus der Verbleibenden Auswahl empfinde ich das Partitionierungsmodell als umständlich da für eine Projektion mit allen Attributen mehrere Verbunde notwendig sind. So habe ich mich also für das Basisrelationenmodell entschieden:

Fahrzeugmodell: {[ name: string, sitze: int, erscheinungsjahr: int ]}

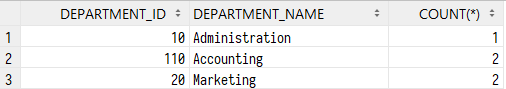
Nutzfahrzeug: {[ name: string, sitze: int, erscheinungsjahr: int, ladefläche: int ]}

Fahrzeugmodell: {[ name: string, sitze: int, erscheinungsjahr: int, kindersitze: int ]}

Entertainment: {[ bezeichung: string, listenpreis: string, webconnect: int ]}

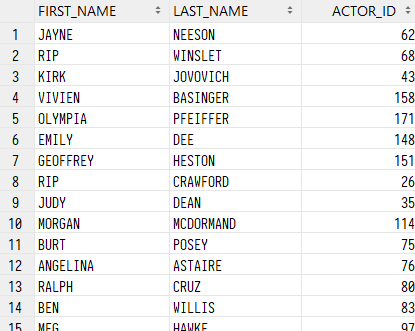
hat: {[ name: string, bezeichnung: string ]}

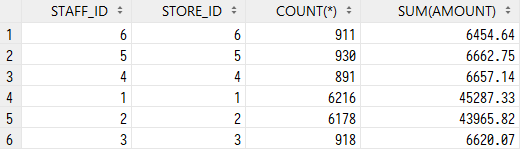
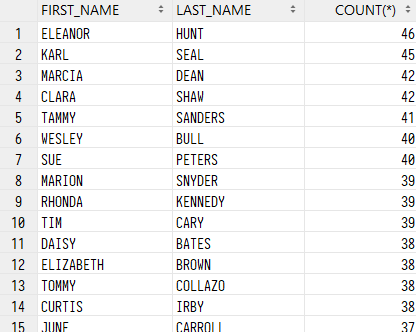
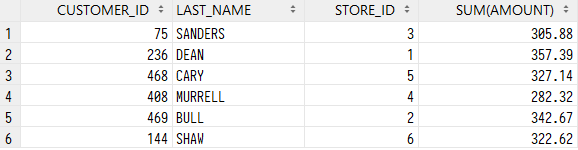
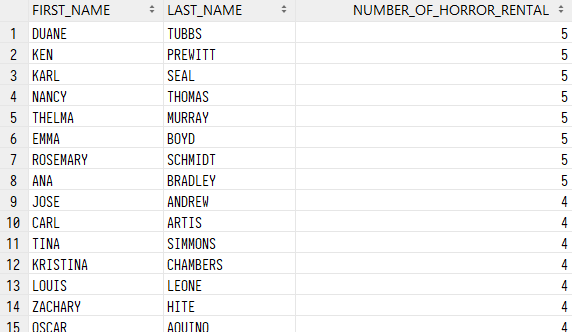
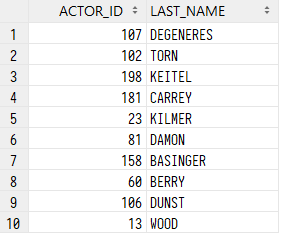
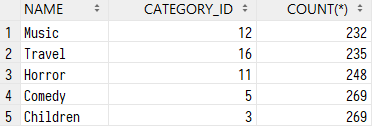
# Aggregate und Gruppierungen

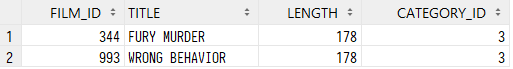
*-- 2 a***SELECT department\_id**, **department\_name**, *Count*(*\**)  
**FROM** departments  
 **INNER JOIN** employees **USING** (**department\_id**)  
**GROUP BY department\_id**, **department\_name  
HAVING** *Count*(*\**) < 3;  
  
*-- 2 b***SELECT department\_id**, **department\_name**, *Count*(*\**)  
**FROM** departments  
 **INNER JOIN** employees **USING** (**department\_id**)  
**GROUP BY department\_id**, **department\_name  
HAVING** *Count*(*\**) = (**SELECT** *MAX*(*COUNT*(**employee\_id**))  
 **FROM** departments  
 **INNER JOIN** employees **USING** (**department\_id**)  
 **GROUP BY department\_id**);  
  
*-- 2 c***SELECT department\_id**, **department\_name**, *Count*(*\**)  
**FROM** departments  
 **INNER JOIN** employees **USING** (**department\_id**)  
**GROUP BY department\_id**, **department\_name  
HAVING** *Count*(*\**) = (**SELECT** *MIN*(*COUNT*(**employee\_id**))  
 **FROM** departments  
 **INNER JOIN** employees **USING** (**department\_id**)  
 **GROUP BY department\_id**);



# Aggregate und Gruppierungen

*-- 3.1 Assumption: The exercise asks for actors who played in multiple films***SELECT first\_name**, **last\_name**, **actor\_id  
FROM** actor  
 **INNER JOIN** film\_actor **USING** (**actor\_id**)  
**GROUP BY first\_name**, **last\_name**, **actor\_id  
HAVING** (*COUNT*(*\**) > 1);  
  
*-- 3.2***SELECT title  
FROM** film  
**WHERE film\_id NOT IN** (**SELECT film\_id FROM** inventory);  
  
*-- 3.3***SELECT r**.**staff\_id**, **s**.**store\_id**, *COUNT*(*\**), *SUM*(**amount**)  
**FROM** rental **r  
 INNER JOIN** payment **USING** (**rental\_id**)  
 **INNER JOIN** staff **s ON** (**s**.**staff\_id** = **r**.**staff\_id**)  
**GROUP BY r**.**staff\_id**, **s**.**store\_id**;

  
*-- 3.4***SELECT first\_name**, **last\_name**, *COUNT*(*\**)  
**FROM** customer  
 **INNER JOIN** rental **USING** (**customer\_id**)  
**GROUP BY first\_name**, **last\_name  
ORDER BY** *COUNT*(*\**) **DESC**;  
  
*-- 3.5***SELECT customer\_id**, **last\_name**, **store\_id**, *SUM*(**amount**)  
**FROM** payment **INNER JOIN** customer c1 **USING** (**customer\_id**)  
**GROUP BY customer\_id**, **last\_name**, **store\_id  
HAVING** *SUM*(**amount**) >= **ALL** (**SELECT** *SUM*(**amount**)  
 **FROM** payment **INNER JOIN** customer c2 **USING** (**customer\_id**)  
 **WHERE** c1.**store\_id** = c2.**store\_id  
 GROUP BY customer\_id**, **last\_name**, **store\_id**);  
  
*-- 3.6***SELECT first\_name**, **last\_name**, *COUNT*(*\**) **AS** number\_of\_horror\_rental  
**FROM** customer  
 **INNER JOIN** rental **USING** (**customer\_id**)  
 **INNER JOIN** inventory **USING** (**inventory\_id**)  
 **INNER JOIN** film\_category **USING** (**film\_id**)  
 **INNER JOIN category USING** (**category\_id**)  
**WHERE name** = **'Horror'  
GROUP BY first\_name**, **last\_name  
HAVING** *COUNT*(*\**) >= 4  
**ORDER BY** *COUNT*(*\**) **DESC**;  
  
*-- 3.7***SELECT actor\_id**, **last\_name  
FROM** actor  
 **INNER JOIN** film\_actor **USING** (**actor\_id**)  
**GROUP BY actor\_id**, **last\_name  
ORDER BY** *COUNT*(*\**) **DESC  
FETCH FIRST** 10 **ROWS ONLY**;  
  
*-- 3.8***SELECT name**, **category\_id**, *COUNT*(*\**)  
**FROM category  
 INNER JOIN** film\_category **USING** (**category\_id**)  
 **INNER JOIN** film **USING** (**film\_id**)  
 **INNER JOIN** inventory **USING** (**film\_id**)  
**GROUP BY category\_id**, **name  
ORDER BY** *COUNT*(*\**) **ASC  
FETCH FIRST** 5 **ROWS ONLY**;  
  
*-- 3.9***SELECT film\_id**, **title**, **length**, **category\_id  
FROM** film  
 **INNER JOIN** film\_category **USING** (**film\_id**)  
**WHERE** (**category\_id**, **length**) **IN** (**SELECT category\_id**, *MAX*(**length**)  
 **FROM** film  
 **INNER JOIN** film\_category fc **USING** (**film\_id**)  
 **GROUP BY category\_id**)  
**ORDER BY length ASC  
FETCH FIRST** 1 **ROWS WITH TIES**;



# GROUP BY und ROLLUP

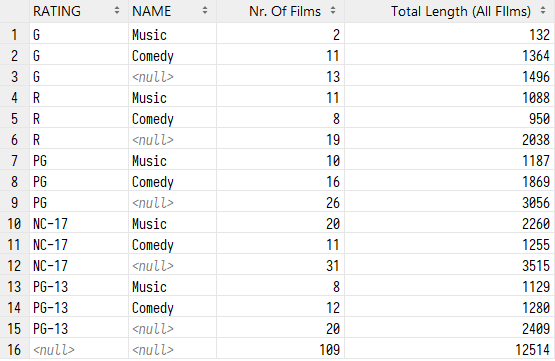
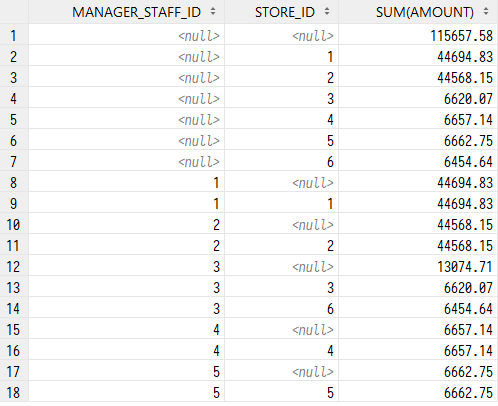
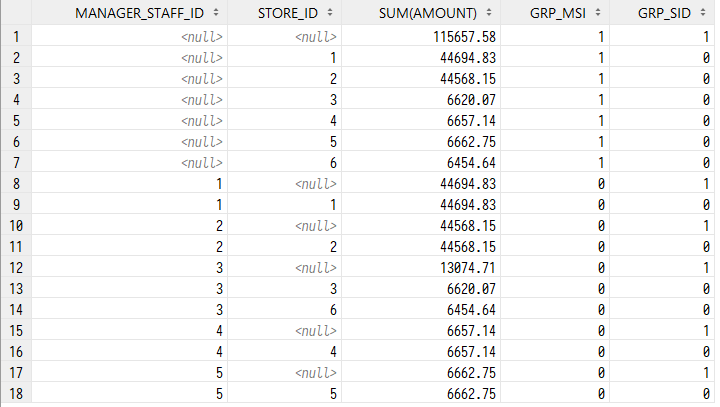
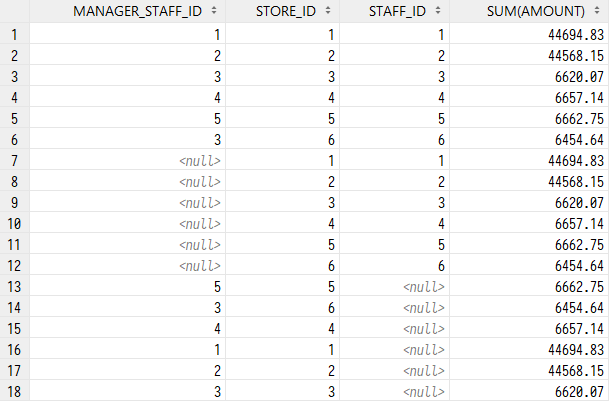


Figure 1 Ergebnis für alle drei Abfragen

*-- 4.1***SELECT rating**, **name**, *COUNT*(*\**) **AS** "Nr. Of Films", *SUM*(**length**) **AS** "Total Length (All FIlms)"  
**FROM** film  
 **INNER JOIN** film\_category **USING** (**film\_id**)  
 **INNER JOIN category USING** (**category\_id**)  
**WHERE name IN**(**'Comedy'**, **'Music'**)  
**GROUP BY ROLLUP** (**rating**, **name**);  
  
*-- 4.2***SELECT rating**, **name**, *COUNT*(*\**) **AS** "Nr. Of Films", *SUM*(**length**) **AS** "Total Length (All FIlms)"  
**FROM** film  
 **INNER JOIN** film\_category **USING** (**film\_id**)  
 **INNER JOIN category USING** (**category\_id**)  
**WHERE name IN**(**'Comedy'**, **'Music'**)  
**GROUP BY GROUPING SETS** ( (**rating**),  
 (**name**, **rating**),  
 ());  
  
*-- 4.3***SELECT rating**, **name**, *COUNT*(*\**) **AS** "Nr. Of Films", *SUM*(**length**) **AS** "Total Length (All Films)"  
**FROM** film  
 **INNER JOIN** film\_category **USING** (**film\_id**)  
 **INNER JOIN category USING** (**category\_id**)  
**WHERE name IN**(**'Comedy'**, **'Music'**)  
**GROUP BY rating**, **name  
UNION ALL  
SELECT rating**, **NULL AS name**, *COUNT*(*\**) **AS** "Nr. Of Films", *SUM*(**length**) **AS** "Total Length (All Films)"  
**FROM** film  
 **INNER JOIN** film\_category **USING** (**film\_id**)  
 **INNER JOIN category USING** (**category\_id**)  
**WHERE name IN**(**'Comedy'**, **'Music'**)  
**GROUP BY rating**, **NULL  
UNION ALL  
SELECT NULL AS** rating, **NULL AS name**, *COUNT*(*\**) **AS** "Nr. Of Films", *SUM*(**length**) **AS** "Total Length (All Films)"  
**FROM** film  
 **INNER JOIN** film\_category **USING** (**film\_id**)  
 **INNER JOIN category USING** (**category\_id**)  
**WHERE name IN**(**'Comedy'**, **'Music'**);

# GROUP BY mit CUBE

*-- 5.1***SELECT manager\_staff\_id**, **store\_id**, *SUM*(**amount**)  
**FROM store s  
 INNER JOIN** staff **USING** (**store\_id**)  
 **INNER JOIN** payment **USING** (**staff\_id**)  
**GROUP BY CUBE** (**manager\_staff\_id**, **store\_id**);  
  
*-- 5.2***SELECT manager\_staff\_id**, **store\_id**, *SUM*(**amount**),  
 *GROUPING*(**manager\_staff\_id**) grp\_msi,  
 *GROUPING*(**store\_id**) grp\_sid  
**FROM store s  
 INNER JOIN** staff **USING** (**store\_id**)  
 **INNER JOIN** payment **USING** (**staff\_id**)  
**GROUP BY CUBE** (**manager\_staff\_id**, **store\_id**);  
  
*-- 5.3***SELECT manager\_staff\_id**, **store\_id**, **staff\_id**, *SUM*(**amount**)  
**FROM store s  
 INNER JOIN** staff **USING** (**store\_id**)  
 **INNER JOIN** payment **USING** (**staff\_id**)  
**GROUP BY GROUPING SETS** ((**manager\_staff\_id**, **store\_id**, **staff\_id**),  
 (**manager\_staff\_id**, **store\_id**),  
 (**store\_id**, **staff\_id**));  
  
*-- 5.4***SELECT country**, *EXTRACT*(**YEAR FROM payment\_date**), **r**.**staff\_id**, *sum*(**amount**), *count*(*\**)  
**FROM** payment  
 **INNER JOIN** rental **r USING** (**rental\_id**)  
 **INNER JOIN** staff **s ON r**.**staff\_id** = **s**.**staff\_id  
 INNER JOIN** inventory **i USING** (**inventory\_id**)  
 **INNER JOIN store** sto **ON** sto.**store\_id** = **i**.**store\_id  
 INNER JOIN** address **a ON** sto.**address\_id** = **a**.**address\_id  
 INNER JOIN** city **USING** (**city\_id**)  
 **INNER JOIN** country **USING** (**country\_id**)  
**GROUP BY GROUPING SETS** ( (**country\_id**, **country**, *EXTRACT*(**YEAR FROM payment\_date**)),  
 (**r**.**staff\_id**, *EXTRACT*(**YEAR FROM payment\_date**)),  
 ());

