DES3UE WS 2021 Übung 3

Abgabetermin: 27.10.2021

☐ **DES3UEG1: Niklas** **Name** Angelos Angelis **Aufwand in h** 5
☒ **DES3UEG2: Niklas**
☐ **DES3UEG3: Traxler** **Punkte** _____ **Kurzzeichen Tutor** _____

Hinweise und Richtlinien:

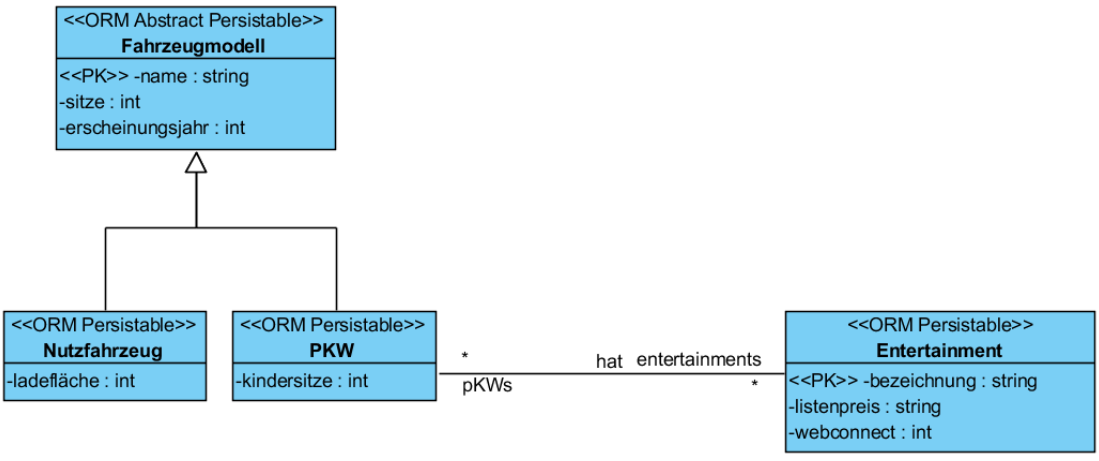
- Übungsausarbeitungen müssen den im eLearning angegebenen Formatierungsrichtlinien entsprechen – Nichtbeachtung dieser Formatierungsrichtlinien führt zu Punkteabzug.
- Zusätzlich zu den allgemeinen Formatierungsrichtlinien sind für diese Übungsausarbeitung folgende zusätzlichen Richtlinien zu beachten:
  - Treffen Sie, falls notwendig, sinnvolle Annahmen und dokumentieren Sie diese nachvollziehbar in ihrer Lösung!
  - Recherchieren Sie eventuell unbekannte Elemente nach Bedarf.

Ziel dieser Übung ist es, ausgewählte Bereiche der Anfragesprache SQL zu wiederholen und zu vertiefen. Durch komplexe Beispiele wird demonstriert wie einzelne SQL Konstrukte kombiniert werden können. Verwenden Sie für die Ausarbeitung je nach Aufgabenstellung die zur Verfügung gestellte HR-Datenbank bzw. die Sakila-Datenbank.

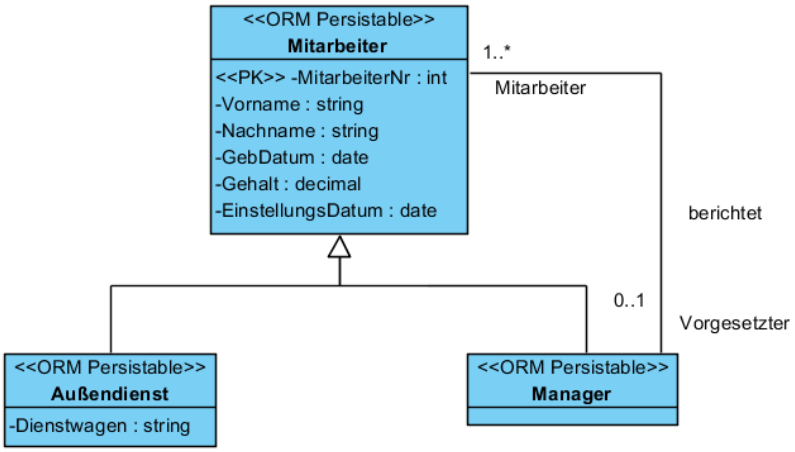
1. Abbildung Generalisierung **(5 Punkte** – je 2,5 Pkte.**)**

Erstellen Sie für die angegebenen UML-Klassen-Diagramme jeweils ein **Relationenmodell**, um die Attribute in Tabellen in der Datenbank abzubilden, zB Tabelle 1 (PKAttr1, Attr2). Wählen Sie jeweils ein geeignetes Abbildungsmodell für die Generalisierungsbeziehung. Nennen Sie die Vor- und Nachteile des gewählten Abbildungsmodells und nehmen Sie dabei Bezug auf die Eigenschaften der Generalisierungsbeziehung, d.h. ob diese vollständig/unvollständig und überlappend/disjunkt ist.

a) Ein Fahrzeughersteller verwaltet die Fahrzeuge von zwei Sparten Nutzfahrzeuge und PKWs. Die Sparten werden getrennt verwaltet, ein Modell wird nicht in beiden Sparten hergestellt.



b) In einem Unternehmen werden Mitarbeiter verwaltet, Mitarbeiter im Außendienst erhalten ein Fahrzeug zur Verfügung gestellt; im Außendienst gibt es ebenso Manager.



2. Aggregate und Gruppierungen (Human Resources) (3 Punkte)

1. Erstellen Sie die folgenden Statistikberichte für die Personalabteilung: Nehmen Sie die Abteilungsnummer, den Namen und die Anzahl der Angestellten für jede Abteilung auf, die folgende Bedingungen erfüllt:

    a. Keine oder mehr als 3 Angestellte: 1 Punkt
    b. Höchste Anzahl von Angestellten: 1 Punkt
    c. Niedrigste Anzahl von Angestellten: 1 Punkt

    Beachten Sie dabei, dass es auch Abteilungen ohne Mitarbeiter geben kann.

3. Aggregate und Gruppierungen (Sakila-Datenbank) (9 Punkte)

1. Welche Schauspieler/Schauspielerinnen (Vor- und Nachname) sind in der Datenbank öfters als einmal eingetragen? (0,5 Punkt)
2. Ermitteln Sie die Titel jener Filme, die zwar in der Filmdatenbank existieren, allerdings in keinem Geschäft angeboten werden. (*42 Zeilen*) (1 Punkt)
3. Geben Sie an, welcher Verkäufer in welchem Geschäft wie viele Verleihungen durchgeführt hat und wie viel Umsatz er (jeweils) erzielt hat. (1 Punkt)

    Achtung: Der Umsatz zählt für den Verkäufer, der den Film verliehen hat, d.h. der die Verleihung eines Films durchgeführt hat. Dieser kann sich vom Verkäufer unterscheiden, der den Bezahlvorgang (payment) durchgeführt hat.
4. Geben Sie pro Kunde (Vorname, Nachname) die Summe geliehener Filme an. Sortieren Sie nach der Anzahl aufsteigend. (1 Punkt)
5. Geben Sie den besten Kunden (gemessen am Umsatz, d.h. wie viel er gesamt bezahlt hat) pro Store an, sowie den Umsatz und die Store-ID des Kunden). Bedenken Sie dabei den Fall, dass Personen gleich heißen könnten. (1,5 Punkte)
6. Geben Sie in Absteigender Reihenfolge an, welcher Kunde (Vorname, Nachname) bereits vier oder mehr Horror-Filme ausgeliehen hat. (*34 Zeilen*) (1,5 Punkte)
7. Geben Sie die Top 10 der Schauspieler aus (mit Vor- und Nachnamen), die in den meisten Filmen mitgespielt haben. (*10 Zeilen, Stichprobe, keine Ties berücksichtigen*) (1 Punkt)

8. Bestimmen Sie den/die kürzeste(n) Film(e) (Titel ausgeben), der der längste Film einer Kategorie ist, zu der er gehört. (*exakt, kein ROWNUM*) (1,5 Punkte)

4. GROUP BY mit ROLLUP (Sakila-Datenbank) (3 Punkte)

Erstellen Sie eine Filmabfrage und ermitteln Sie die Anzahl der Filme und die Summe der Längen in den Kategorien ‚Comedy' und ‚Music'

- je Rating
- je Kategorie und Rating
- „Comedy' und ‚Music' gesamt

Entwickeln Sie drei Varianten, einmal mit der Verwendung des GROUP BY ROLLUP Operators (1 Punkt) und zum Vergleich dazu mit GROUPING SETS (1 Punkt) und komplett ohne ROLLUP und SETS (Hinweis: 3 Selects mit UNION) (1 Punkt). Vergleichen Sie die Zugriffspläne und kommentieren Sie diese.

Hinweis: SET AUTOTRACE ON erlaubt in SQL*PLUS die Ausgabe des Zugriffsplan sowie einiger Statistiken eines SQL-Statement und bietet eine einfache Möglichkeit zum Vergleich von Abfragen.

5. Group By mit CUBE (Sakila-Datenbank) (4 Punkte)

1. Erstellen Sie mit `GROUPING SETS` eine Abfrage, um folgende Gruppierungen anzuzeigen:
   - `manager_staff_id, store_id, staff_id`
   - `manager_staff_id, store_id`
   - `store_id, staff_id`

   Die Abfrage soll die Summe der Erlöse für jede dieser Gruppen berechnen. (1 Punkt)

(Auszug aus der Lösung; Bitte vervollständigen Sie die Tabelle in Ihrer Abgabe)

MANAGER_STAFF_ID	STORE_ID	STAFF_ID	Gesamterlös
1	1	1	44694,83
2	2	2	…
…	…		…
5	5	5	6662,75
3	6	6	…
…	…	…	…
1	1	(null)	44694,83
3	…	(null)	6620,07
(null)	1	1	44694,83
…	…	…	…
(null)	6	6	6454,64

2. Erstellen Sie einen Bericht über verliehene Filme, der pro Land, Jahr und Kategorie die Summe des Umsatzes und die Anzahl der Bezahlvorgänge (vgl. payment) zusammenfasst. Berücksichtigen Sie nur die Kategorien 'Family', 'Children' und 'Travel'. Erstellen Sie ein SQL-Statement und verwenden Sie GROUPING SETS. Sortieren Sie die Ausgabe nach Ländern. (1,5 Punkte)

(Auszug aus der Lösung 5.2; Bitte vervollständigen Sie die Tabelle in Ihrer Abgabe)

JAHR	LAND	KATEGORIE	ANZAHL	SUMME
2013	Australia	(null)	5	41,16
2013	Austria	(null)	15	114,45
…	…	…	…	…
(null)	Australia	Children	153	1128,88
…	…	…	…	…
(null)	Austria	Children	174	1205,44
…	…	…	…	…
(null)	(null)	(null)	2878	21871,51

3. Erstellen Sie eine Abfrage, um für alle Manager, die in den Stores angestellt sind, folgendes anzuzeigen (1 Punkte):
   - Manager-Id
   - Store und Gesamterlös für jeden Manager
   - Gesamterlöse aller Manager
   - Kreuztabellenwerte für die Anzeige des Gesamterlöses für jeden Standort
   - Gesamterlös, unabhängig vom Standort

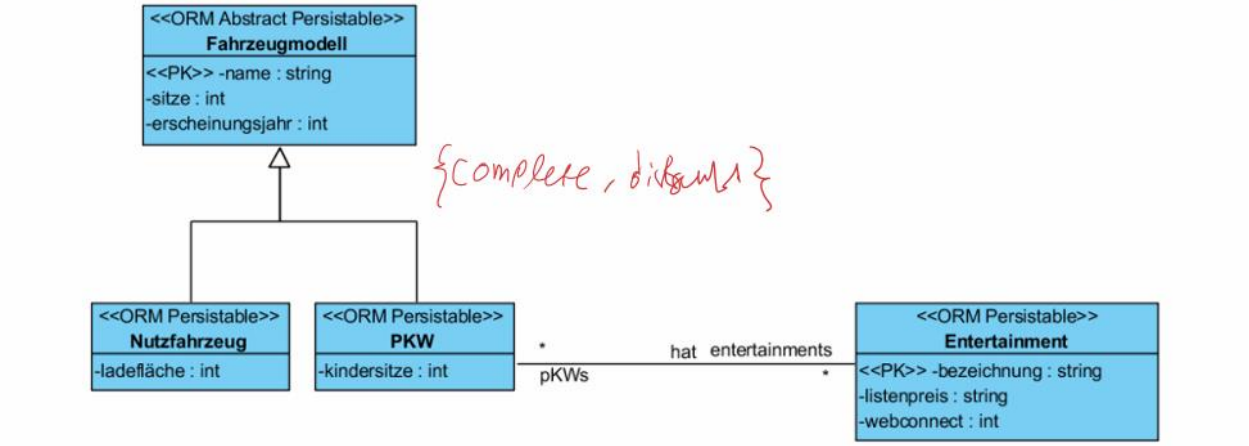
(Auszug aus der Lösung; Bitte vervollständigen Sie die Tabelle in Ihrer Abgabe)

MANAGER-ID	STORE	Gesamterlös
(null)	(null)	115657,58
(null)	1	44694,83
…	…	…
…	…	…
3	(null)	13074,71
3	3	6620,07
3	6	6454,64
…	…	…

4. Prüfen Sie die Ausgabe der obigen Aufgabe. Schreiben Sie mit der `GROUPING`-Funktion eine Abfrage, um festzustellen, ob die Nullwerte in den Spalten, die den `GROUP BY` Ausdrücken entsprechen, von der `CUBE`-Operation verursacht werden. (0,5 Punkte)

Aufgabe 1)

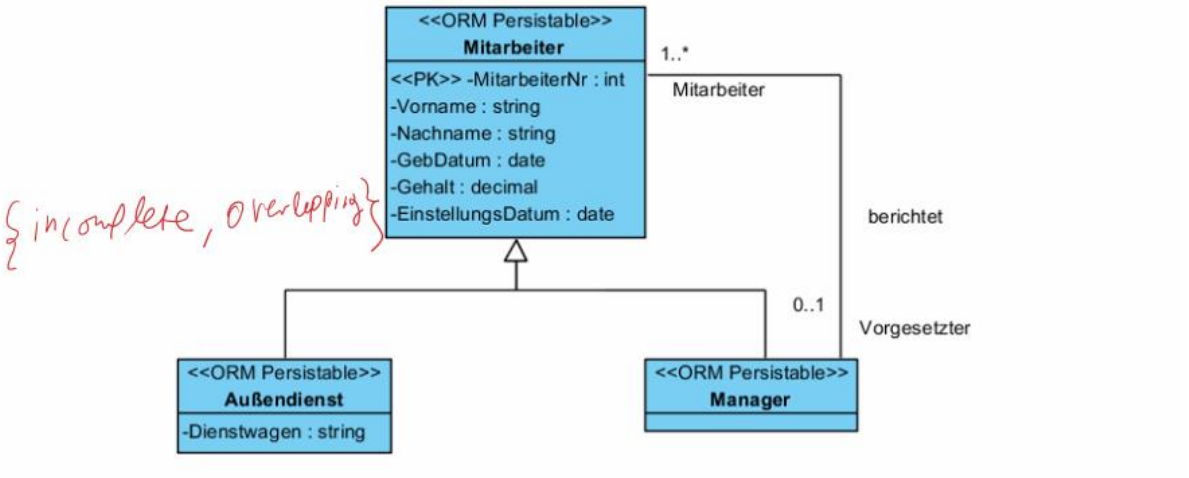
a) Ein Fahrzeughersteller verwaltet die Fahrzeuge von zwei Sparten Nutzfahrzeuge und PKWs. Die Sparten werden getrennt verwaltet, ein Modell wird nicht in beiden Sparten hergestellt.



FAHRZEUGMODELL(Name: String; size: Int; Erscheinungsjahr:int; ladefläche:int; kindersitze:int; Fahrzeugmodelltyp:String)
Hat(Name:String; Bezeichnung:String)
Entertainment(bezeichnung:String; listenpreis:String; webconnect: int)

Ich würde hierfür das disjunkte Einrealtionenmodell nutzen. Vorteil hierbei ist dass man einfachen Zugriff auf alle Objekte und Attribute der Klassen hat. Ebenfalls ist das Einfügen und erneuern der Datensätze im Einrelationenmodell viel einfacher. Einziger Nachteil sind die ganzen Null Werte die entstehen werden.

b) In einem Unternehmen werden Mitarbeiter verwaltet, Mitarbeiter im Außendienst erhalten ein Fahrzeug zur Verfügung gestellt; im Außendienst gibt es ebenso Manager.



Mitarbeiter(MitarbeiterNr: Int; Vorname: String; Nachname: String; GebDatum: date; Gehalt: Int; Einstellungsdatum: date; AußendiesntTyp: Boolean; Dienstawagen: String; ManagerTyp: Boolean; Manager.MitarbeiterNr: Int)

Ich würde hierfür das überlappende Einrealtionenmodell nutzen. Vorteil hierbei ist dass man einfachen Zugriff auf alle Objekte und Attribute der Klassen hat. Ebenfalls ist das Einfügen und erneuern der Datensätze im Einrelationenmodell viel einfacher. Einziger Nachteil sind die ganzen Null Werte die entstehen werden.

Aufgabe2)

```
/*1. Erstellen Sie die folgenden Statistikberichte für die Personalabteilung:
Nehmen Sie die
Abteilungsnummer, den Namen und die Anzahl der Angestellten für jede
Abteilung auf, die
folgende Bedingungen erfüllt:
a. Keine oder mehr als 3 Angestellte: 1 Punkt
b. Höchste Anzahl von Angestellten: 1 Punkt
c. Niedrigste Anzahl von Angestellten: 1 Punkt
Beachten Sie dabei, dass es auch Abteilungen ohne Mitarbeiter geben kann*/
SELECT D.DEPARTMENT_ID,D.DEPARTMENT_NAME, COUNT(E.EMPLOYEE_ID)
FROM DEPARTMENTS D
LEFT JOIN EMPLOYEES E on D.DEPARTMENT_ID = E.DEPARTMENT_ID
GROUP BY  D.DEPARTMENT_ID, D.DEPARTMENT_NAME
HAVING (COUNT(E.EMPLOYEE_ID) = 0 OR COUNT(E.EMPLOYEE_ID) > 3);
```

	DEPARTMENT_ID	DEPARTMENT_NAME	"COUNT(E.EMPLOYEE_ID)"
1	190	Contracting	0
2	50	Shipping	5

```
SELECT D.DEPARTMENT_ID,D.DEPARTMENT_NAME,COUNT(E.EMPLOYEE_ID)
FROM DEPARTMENTS D
JOIN EMPLOYEES E on D.DEPARTMENT_ID = E.DEPARTMENT_ID
GROUP BY D.DEPARTMENT_ID, D.DEPARTMENT_NAME
HAVING COUNT(E.EMPLOYEE_ID) = (SELECT MAX(COUNT(E2.EMPLOYEE_ID))
                               FROM DEPARTMENTS D2
                               JOIN EMPLOYEES E2 on D2.DEPARTMENT_ID =
E2.DEPARTMENT_ID
                               GROUP BY D2.DEPARTMENT_ID);
```

	DEPARTMENT_ID	DEPARTMENT_NAME	"COUNT(E.EMPLOYEE_ID)"
1	50	Shipping	5

```
SELECT D.DEPARTMENT_ID,D.DEPARTMENT_NAME,COUNT(E.EMPLOYEE_ID)
FROM DEPARTMENTS D
LEFT JOIN EMPLOYEES E on D.DEPARTMENT_ID = E.DEPARTMENT_ID
GROUP BY D.DEPARTMENT_ID, D.DEPARTMENT_NAME
HAVING COUNT(E.EMPLOYEE_ID) = (SELECT MIN(COUNT(E2.EMPLOYEE_ID))
                               FROM DEPARTMENTS D2
                               LEFT JOIN EMPLOYEES E2 on D2.DEPARTMENT_ID =
E2.DEPARTMENT_ID
                               GROUP BY D2.DEPARTMENT_ID);
```

	DEPARTMENT_ID	DEPARTMENT_NAME	"COUNT(E.EMPLOYEE_ID)"
1	50	Shipping	5

```
-- 1. Welche Schauspieler/Schauspielerinnen (Vor- und Nachname) sind in der Datenbank öfters als
-- einmal eingetragen? (0,5 Punkt)
SELECT FIRST_NAME,LAST_NAME,COUNT(*)
FROM ACTOR
GROUP BY FIRST_NAME, LAST_NAME
HAVING COUNT(*) > 1;
```

	FIRST_NAME	LAST_NAME	"COUNT(*)"
1	SUSAN	DAVIS	2

```
-- 2. Ermitteln Sie die Titel jener Filme, die zwar in der Filmdatenbank existieren, allerdings in keinem
-- Geschäft angeboten werden. (42 Zeilen) (1 Punkt)
SELECT F.TITLE
FROM FILM F
LEFT JOIN INVENTORY I on F.FILM_ID = I.FILM_ID
GROUP BY F.TITLE
HAVING COUNT(STORE_ID) = 0;
```

	TITLE		
1	COMMANDMENTS EXPRESS	38	DAZED PUNK
2	FRANKENSTEIN STRANGER	39	GLADIATOR WESTWARD
3	RAINBOW SHOCK	40	ARGONAUTS TOWN
4	TADPOLE PARK	41	KENTUCKIAN GIANT
5	SUICIDES SILENCE	42	BOONDOCK BALLROOM

```
-- 3. Geben Sie an, welcher Verkäufer in welchem Geschäft wie viele Verleihungen durchgeführt hat
-- und wie viel Umsatz er (jeweils) erzielt hat. (1 Punkt)
-- Achtung: Der Umsatz zählt für den Verkäufer, der den Film verliehen hat, d.h. der die Verleihung
-- eines Films durchgeführt hat. Dieser kann sich vom Verkäufer unterscheiden, der den
-- Bezahlvorgang (payment) durchgeführt hat.
SELECT S.FIRST_NAME,S.LAST_NAME,S.STORE_ID, COUNT(R.RENTAL_ID),SUM(P.AMOUNT)
FROM STAFF S
LEFT JOIN RENTAL R on S.STAFF_ID = R.STAFF_ID
LEFT JOIN PAYMENT P on R.RENTAL_ID = P.RENTAL_ID
GROUP BY S.FIRST_NAME, S.LAST_NAME, S.STORE_ID;
```

	FIRST_NAME	LAST_NAME	STORE_ID	"COUNT(R.RENTAL_ID)"	"SUM(P.AMOUNT)"
1	Tracey	Hynf	4	891	6657.14
2	Mary	Brown	5	930	6662.75
3	Stephen	Miller	3	918	6620.07
4	Mike	Hillyer	1	6216	45287.33
5	Frank	Delfino	6	911	6454.64
6	Jon	Stephens	2	6178	43965.82

```
-- 4. Geben Sie pro Kunde (Vorname, Nachname) die Summe geliehener Filme an. Sortieren Sie nach
-- der Anzahl aufsteigend. (1 Punkt)
SELECT C.FIRST_NAME,C.LAST_NAME,COUNT(RENTAL_ID)
FROM CUSTOMER C
LEFT JOIN RENTAL R on C.CUSTOMER_ID = R.CUSTOMER_ID
GROUP BY C.FIRST_NAME, C.LAST_NAME
ORDER BY COUNT(RENTAL_ID) ASC ;
```

	FIRST_NAME	LAST_NAME	"COUNT(RENTAL_ID)"
1	BRIAN	WYMAN	12
2	TIFFANY	JORDAN	14
3	KATHERINE	RIVERA	14
4	LEONA	OBRIEN	14
5	CAROLINE	BOWMAN	15
495	SYLVIA	ORTIZ	32
496	LEONARD	SCHOFIELD	32
497	ARTHUR	SIMPKINS	32
498	VERONICA	STONE	32
499	COURTNEY	DAY	32
500	LEROY	BUSTAMANTE	32

```
-- 5. Geben Sie den besten Kunden (gemessen am Umsatz, d.h. wie viel er
gesamt bezahlt hat) pro
-- Store an, sowie den Umsatz und die Store-ID des Kunden). Bedenken Sie
dabei den Fall, dass
-- Personen gleich heißen könnten. (1,5 Punkte)
SELECT C.FIRST_NAME, C.LAST_NAME, C.STORE_ID,SUM(P.AMOUNT) AS "UMSATZ"
FROM CUSTOMER C
JOIN RENTAL R on C.CUSTOMER_ID = R.CUSTOMER_ID
JOIN PAYMENT P on R.RENTAL_ID = P.RENTAL_ID
GROUP BY C.FIRST_NAME, C.LAST_NAME, C.STORE_ID
HAVING SUM(P.AMOUNT) IN (SELECT MAX(UMSATZ)
                         FROM (SELECT STORE_ID,FIRST_NAME,SUM(AMOUNT) AS
"UMSATZ"
                               FROM CUSTOMER
                               INNER JOIN RENTAL USING(CUSTOMER_ID)
                               INNER JOIN PAYMENT USING(RENTAL_ID)
                               GROUP BY STORE_ID, FIRST_NAME)
                         GROUP BY STORE_ID);
```

	FIRST_NAME	LAST_NAME	STORE_ID	UMSATZ
1	TAMMY	SANDERS	3	305.88
2	MARCIA	DEAN	1	357.39
3	TIM	CARY	5	327.14
4	WESLEY	BULL	2	342.67
5	CLARA	SHAW	6	322.62
6	MANUEL	MURRELL	4	282.32

```
-- 6. Geben Sie in Absteigender Reihenfolge an, welcher Kunde (Vorname,
Nachname) bereits vier
-- oder mehr Horror-Filme ausgeliehen hat. (34 Zeilen) (1,5 Punkte)
SELECT C.FIRST_NAME,C.LAST_NAME,COUNT(RENTAL_ID)
FROM CUSTOMER C
JOIN RENTAL R on C.CUSTOMER_ID = R.CUSTOMER_ID
JOIN INVENTORY I on R.INVENTORY_ID = I.INVENTORY_ID
JOIN FILM_CATEGORY FC on I.FILM_ID = FC.FILM_ID
JOIN CATEGORY C2 on FC.CATEGORY_ID = C2.CATEGORY_ID
WHERE C2.NAME = 'Horror'
GROUP BY C.FIRST_NAME, C.LAST_NAME
HAVING COUNT(RENTAL_ID) > 3;
```

	FIRST_NAME	LAST_NAME	"COUNT(RENTAL_ID)"
1	SUE	PETERS	4
2	RICHARD	MCCRARY	4
3	CARL	ARTIS	4
4	STEPHEN	QUALLS	4
5	CONNIE	WALLACE	4
30	RAYMOND	MCWHORTER	4
31	SONIA	GREGORY	4
32	ELEANOR	HUNT	4
33	OSCAR	AQUINO	4
34	TINA	SIMMONS	4

```
-- 7. Geben Sie die Top 10 der Schauspieler aus (mit Vor- und Nachnamen), die
in den meisten Filmen
-- mitgespielt haben. (10 Zeilen, Stichprobe, keine Ties berücksichtigen) (1
Punkt)
SELECT A.FIRST_NAME,A.LAST_NAME,COUNT(FA.FILM_ID)
FROM ACTOR A
JOIN FILM_ACTOR FA on A.ACTOR_ID = FA.ACTOR_ID
```

```
GROUP BY A.FIRST_NAME, A.LAST_NAME
ORDER BY COUNT(FA.FILM_ID) DESC fetch first 10 rows only;
```

	FIRST_NAME	LAST_NAME	"COUNT(FA.FILM_ID)"
2	GINA	DEGENERES	42
3	WALTER	TORN	41
4	MARY	KEITEL	40
5	MATTHEW	CARREY	39
6	SANDRA	KILMER	37
7	SCARLETT	DAMON	36
8	VIVIEN	BASINGER	35
9	UMA	WOOD	35
10	ANGELA	WITHERSPOON	35

```
-- 8. Bestimmen Sie den/die kürzeste(n) Film(e) (Titel ausgeben), der der längste Film einer Kategorie
-- ist, zu der er gehört. (exakt, kein ROWNUM)
SELECT F.TITLE
FROM FILM F
JOIN FILM_CATEGORY FC on F.FILM_ID = FC.FILM_ID
WHERE LENGTH
```

Aufgabe4)

```
SELECT NAME, RATING, COUNT(*), SUM(LENGTH)
FROM FILM
JOIN FILM_CATEGORY USING (FILM_ID)
JOIN CATEGORY USING (CATEGORY_ID)
WHERE NAME = 'Comedy' OR NAME = 'Music'
GROUP BY ROLLUP(NAME,RATING);
```

Total Cost
16.0

	NAME	RATING	"COUNT(*)"	"SUM(LENGTH)"
1	Music	G	2	132
2	Music	R	11	1088
3	Music	PG	10	1187
4	Music	NC-17	20	2260
5	Music	PG-13	8	1129
6	Music	<null>	51	5796
7	Comedy	G	11	1364
8	Comedy	R	8	950
9	Comedy	PG	16	1869
10	Comedy	NC-17	11	1255
11	Comedy	PG-13	12	1280
12	Comedy	<null>	58	6718
13	<null>	<null>	109	12514

```
SELECT NAME,RATING,COUNT(*),SUM(LENGTH)
FROM FILM
JOIN FILM_CATEGORY USING (FILM_ID)
JOIN CATEGORY USING (CATEGORY_ID)
WHERE NAME = 'Comedy' OR NAME = 'Music'
GROUP BY NAME, RATING
UNION ALL
SELECT NAME, NULL,COUNT(*),SUM(LENGTH)
FROM FILM
JOIN FILM_CATEGORY USING (FILM_ID)
JOIN CATEGORY USING (CATEGORY_ID)
WHERE NAME = 'Comedy' OR NAME = 'Music'
GROUP BY NAME
UNION ALL
SELECT NULL, NULL,COUNT(*),SUM(LENGTH)
FROM FILM
JOIN FILM_CATEGORY USING (FILM_ID)
JOIN CATEGORY USING (CATEGORY_ID)
WHERE NAME = 'Comedy' OR NAME = 'Music'
GROUP BY RATING
ORDER BY NAME;
```

	NAME	RATING	"COUNT(*)"	"SUM(LENGTH)"
1	Comedy	R	8	950
2	Comedy	<null>	58	6718
3	Comedy	NC-17	11	1255
4	Comedy	PG-13	12	1280
5	Comedy	G	11	1364
6	Comedy	PG	16	1869
7	Music	PG-13	8	1129
8	Music	PG	10	1187
9	Music	G	2	132
10	Music	NC-17	20	2260
11	Music	R	11	1088
12	Music	<null>	51	5796
13	<null>	<null>	20	2409
14	<null>	<null>	19	2038
15	<null>	<null>	31	3515
16	<null>	<null>	13	1496
17	<null>	<null>	26	3056

Die Laufzeit bei Rollup ist um etwa das zweifache schneller als die Lösung ohne das Rollup. Deshalb sollte man immer wenn es geht den Rollup operator fürs Grupieren benutzen wenn man zwischensummen bilden will und das nicht mit den mehrfachen Selects und Unions lösen.

Aufgabe5)

```
SELECT S.MANAGER_STAFF_ID,S.STORE_ID,ST.STAFF_ID,SUM(P.AMOUNT)
FROM STORE S
JOIN STAFF ST on S.STORE_ID = ST.STORE_ID
JOIN PAYMENT P on ST.STAFF_ID = P.STAFF_ID
GROUP BY GROUPING SETS
((S.MANAGER_STAFF_ID,S.STORE_ID,ST.STAFF_ID),(S.MANAGER_STAFF_ID,S.STORE_ID),
(S.STORE_ID,ST.STAFF_ID))
```

	MANAGER_STAFF_ID	STORE_ID	STAFF_ID	"SUM(P.AMOUNT)"
1	1	1	1	44694.83
2	2	2	2	44568.15
3	3	3	3	6620.07
4	4	4	4	6657.14
5	5	5	5	6662.75
6	3	6	6	6454.64
7	4	4	<null>	6657.14
8	2	2	<null>	44568.15
9	5	5	<null>	6662.75
10	3	6	<null>	6454.64
11	1	1	<null>	44694.83
12	3	3	<null>	6620.07
13	<null>	1	1	44694.83
14	<null>	2	2	44568.15
15	<null>	3	3	6620.07
16	<null>	4	4	6657.14
17	<null>	5	5	6662.75
18	<null>	6	6	6454.64

```
SELECT
F.TITLE,F.RELEASE_YEAR,C3.COUNTRY,C4.NAME,SUM(P.AMOUNT),COUNT(P.PAYMENT_ID)
FROM FILM F
JOIN INVENTORY I on F.FILM_ID = I.FILM_ID
JOIN RENTAL R on I.INVENTORY_ID = R.INVENTORY_ID
JOIN PAYMENT P on R.RENTAL_ID = P.RENTAL_ID
JOIN STORE S on S.STORE_ID = I.STORE_ID
JOIN ADDRESS A2 on A2.ADDRESS_ID = S.ADDRESS_ID
JOIN CITY C2 on C2.CITY_ID = A2.CITY_ID
JOIN COUNTRY C3 on C3.COUNTRY_ID = C2.COUNTRY_ID
JOIN FILM_CATEGORY FC on F.FILM_ID = FC.FILM_ID
JOIN CATEGORY C4 on C4.CATEGORY_ID = FC.CATEGORY_ID
WHERE C4.NAME = 'Family' OR C4.NAME ='Children' OR C4.NAME ='Travel'
GROUP BY GROUPING SETS ( (F.TITLE,F.RELEASE_YEAR,C3.COUNTRY,C4.NAME) )
ORDER BY C3.COUNTRY;
```

	TITLE	RELEASE_YEAR	COUNTRY	NAME	"SUM(P.AMOUNT)"	"COUNT(P.PAYMENT_ID)"
1	COMFORTS RUSH	1989	Australia	Travel	10.75	3
2	SPLASH GUMP	1995	Australia	Family	23.12	2
3	HALF OUTFIELD	1988	Australia	Family	30.67	2
4	SLUMS DUCK	1999	Australia	Family	40.55	4
5	DESPERATE TRAINSPOTTING	1988	Australia	Travel	46.24	3
476	SPARTACUS CHEAPER	1996	Japan	Family	38	5
477	ZOOLANDER FICTION	2002	Japan	Children	10.99	8
478	LABYRINTH LEAGUE	2002	Japan	Children	66.06	7
479	COMFORTS RUSH	1989	Japan	Travel	23.71	4
480	HEARTBREAKERS BRIGHT	2007	Japan	Children	24.99	4

```
SELECT S.MANAGER_STAFF_ID,S.STORE_ID,SUM(AMOUNT)
FROM STORE S
JOIN STAFF S2 on S.STORE_ID = S2.STORE_ID
JOIN PAYMENT P on S2.STAFF_ID = P.STAFF_ID
GROUP BY CUBE(S.STORE_ID,S.MANAGER_STAFF_ID);
```

	MANAGER_STAFF_ID	STORE_ID	"SUM(AMOUNT)"
1	<null>	<null>	115657.58
2	1	<null>	44694.83
3	2	<null>	44568.15
4	3	<null>	13074.71
5	4	<null>	6657.14
6	5	<null>	6662.75
7	<null>	1	44694.83
8	1	1	44694.83
9	<null>	2	44568.15
10	2	2	44568.15
11	<null>	3	6620.07
12	3	3	6620.07
13	<null>	4	6657.14
14	4	4	6657.14
15	<null>	5	6662.75
16	5	5	6662.75
17	<null>	6	6454.64
18	3	6	6454.64

```
SELECT S.MANAGER_STAFF_ID,S.STORE_ID,SUM(AMOUNT),
       GROUPING(S.MANAGER_STAFF_ID) GRP_MANAGER_ID,
       GROUPING(S.STORE_ID) GRP_STORE_ID
FROM STORE S
JOIN STAFF S2 on S.STORE_ID = S2.STORE_ID
JOIN PAYMENT P on S2.STAFF_ID = P.STAFF_ID
GROUP BY CUBE(S.STORE_ID,S.MANAGER_STAFF_ID);
```

	MANAGER_STAFF_ID	STORE_ID	"SUM(AMOUNT)"	GRP_MANAGER_ID	GRP_STORE_ID
1	<null>	<null>	115657.58	1	1
2	1	<null>	44694.83	0	1
3	2	<null>	44568.15	0	1
4	3	<null>	13074.71	0	1
5	4	<null>	6657.14	0	1
6	5	<null>	6662.75	0	1
7	<null>	1	44694.83	1	0
8	1	1	44694.83	0	0
9	<null>	2	44568.15	1	0
10	2	2	44568.15	0	0
11	<null>	3	6620.07	1	0
12	3	3	6620.07	0	0
13	<null>	4	6657.14	1	0
14	4	4	6657.14	0	0
15	<null>	5	6662.75	1	0
16	5	5	6662.75	0	0
17	<null>	6	6454.64	1	0
18	3	6	6454.64	0	0