	Lehrveranstaltung	Databases and Information Systems 2020		
	Aufgabenzettel	1		
	STiNE-Gruppe 14	Simon Weidmann, Aram Yesildeniz		
	Ausgabe	28. April 2020	Abgabe	8. Mai 2020

## 1.2 DB-Schema

### Tables

=> Horizontal Partitioning

Estate Agent			
<u>Login</u>	Name	Address	Password

Estate						
<u>ID</u>	City	Postal Code	Street	Street Number	Square Area	Manager = EstateAgent.Login

Apartment											
<u>ID</u>	City	PC	Str.	Str.Nr.	SA	Floor	Rent	Rooms	Balcony	Kitchen	<u>Manager = EstateAgent.Login</u>

House									
<u>ID</u>	City	PC	Str.	Str.Nr.	SA	Floors	Price	Garden	Manager = EstateAgent.Login

Person			
<u>ID</u>	First Name	Name	Address

Contract		
<u>Contract No.</u>	Date	Place


Tenancy Contract							
<u>Con.No.</u>	Date	Place	Start D.	Dur.	Add.Costs	Person.ID	Apartment.ID

Purchase Contract						
<u>Con.No.</u>	Date	Place	No.of.Install.	Intrest Rate	Person.ID	House.ID

### SQL Scripts

#### Create Database

```
CREATE DATABASE dis
WITH
OWNER = postgres
ENCODING = 'UTF8'
CONNECTION LIMIT = -1;
```

	Lehrveranstaltung	Databases and Information Systems 2020		
	Aufgabenzettel	1		
	STiNE-Gruppe 14	Simon Weidmann, Aram Yesildeniz		
	Ausgabe	28. April 2020	Abgabe	8. Mai 2020

## Create Tables


```
CREATE TABLE public.estate_agent
(
    agent_login text,
    agent_name text,
    agent_address text,
    agent_password text,
    PRIMARY KEY (agent_login)
);
```

```
ALTER TABLE public.estate_agent
    OWNER to postgres;
```

```
CREATE TABLE public.estate
(
    estate_id serial,
    city text,
    postal_code integer,
    street text,
    street_number text,
    square_area integer,
    manager text,
    PRIMARY KEY (estate_id),
    CONSTRAINT manager FOREIGN KEY (manager)
        REFERENCES public.estate_agent (agent_login) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID
);
```

```
ALTER TABLE public.estate
    OWNER to postgres;
```

```
CREATE TABLE public.apartment
(
    floor integer,
    rent text,
    rooms text,
    balcony boolean,
    kitchen boolean,
```

	Lehrveranstaltung	<b>Databases and Information Systems 2020</b>		
	Aufgabenzettel	<b>1</b>		
	STiNE-Gruppe 14	<b>Simon Weidmann, Aram Yesildeniz</b>		
	Ausgabe	<b>28. April 2020</b>	Abgabe	<b>8. Mai 2020</b>

```

CONSTRAINT apartment_pkey PRIMARY KEY (estate_id),
CONSTRAINT manager FOREIGN KEY (manager)
    REFERENCES public.estate_agent (agent_login) MATCH SIMPLE
    ON UPDATE NO ACTION
    ON DELETE NO ACTION
    NOT VALID
)
INHERITS (public.estate);

ALTER TABLE public.apartment
    OWNER to postgres;


CREATE TABLE public.house
(
    floors integer,
    price text,
    garden boolean,
    CONSTRAINT house_pkey PRIMARY KEY (estate_id),
    CONSTRAINT manager FOREIGN KEY (manager)
        REFERENCES public.estate_agent (agent_login) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
)
INHERITS (public.estate)
TABLESPACE pg_default;

ALTER TABLE public.house
    OWNER to postgres;

CREATE TABLE public.person
(
    id serial,
    first_name text,
    last_name text,
    address text,
    PRIMARY KEY (id)
);

ALTER TABLE public.person
    OWNER to postgres;

```

	Lehrveranstaltung	<b>Databases and Information Systems 2020</b>		
	Aufgabenzettel	<b>1</b>		
	STiNE-Gruppe 14	<b>Simon Weidmann, Aram Yesildeniz</b>		
	Ausgabe	<b>28. April 2020</b>	Abgabe	<b>8. Mai 2020</b>


```
CREATE TABLE public.contract
(
    contract_number serial,
    contract_date date,
    place text,
    PRIMARY KEY (contract_number)
);
```

```
ALTER TABLE public.contract
    OWNER to postgres;
```

```
CREATE TABLE public.tenancy_contract
(
    start_date date,
    duration text,
    additional_costs text,
    person_id integer,
    apartment_id integer,
    CONSTRAINT tenancy_contract_pkey PRIMARY KEY (contract_number),
    CONSTRAINT person_id FOREIGN KEY (person_id)
        REFERENCES public.person (id) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT apartment_id FOREIGN KEY (apartment_id)
        REFERENCES public.apartment (estate_id) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID
)
INHERITS (public.contract);
```

```
ALTER TABLE public.tenancy_contract
    OWNER to postgres;
```

```
CREATE TABLE public.purchase_contract
(
    installment_amount text,
    interest_rate text,
    person_id integer,
    house_id integer,
    CONSTRAINT purchase_contract_pkey PRIMARY KEY (contract_number),
```

	Lehrveranstaltung	Databases and Information Systems 2020		
	Aufgabenzettel	1		
	STiNE-Gruppe 14	Simon Weidmann, Aram Yesildeniz		
	Ausgabe	28. April 2020	Abgabe	8. Mai 2020

```

CONSTRAINT person_id FOREIGN KEY (person_id)
    REFERENCES public.person (id) MATCH SIMPLE
    ON UPDATE NO ACTION
    ON DELETE NO ACTION
    NOT VALID,
CONSTRAINT house_id FOREIGN KEY (house_id)
    REFERENCES public.house (estate_id) MATCH SIMPLE
    ON UPDATE NO ACTION
    ON DELETE NO ACTION
    NOT VALID
)
INHERITS (public.contract);

ALTER TABLE public.purchase_contract
    OWNER to postgres;

```

### Insert Estate Agent

```

INSERT INTO public.estate_agent(
    agent_login, agent_name, agent_address, agent_password)
VALUES ('testagent', 'testname', 'testaddress', 'testpassword');

```

## 1.3 Java Application - Questions

*Create an apartment, an estate agent and a tenancy contract with your java application. Validate that they are in the database (e.g. by using a screenshot of application and database).*

*Create a contract with a non-existing estate. Does it work? Why/Why not?*

*Which inheritance model did you choose and why?*

Horizontal: Was more intuitive for us and easier to implement.

*Create an apartment, and let your application crash between inserting the estate information and inserting the apartment information. What is the effect on your database state?*