Veritabanı Yönetimi

BLM3041-1

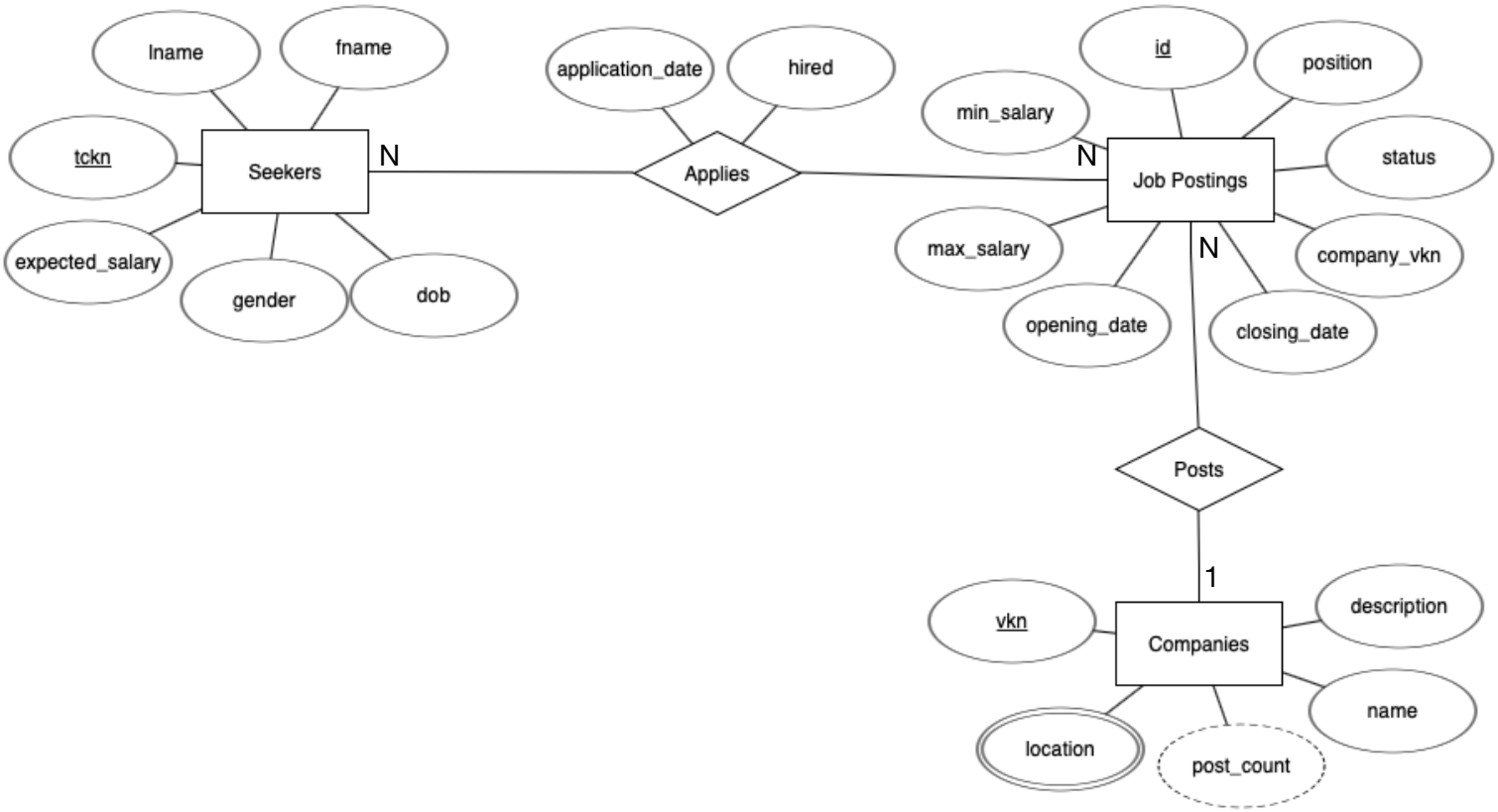
Arş.Gör. Elçin GÜVEYİ

Dr.Öğr.Üyesi Mustafa Utku KALAY

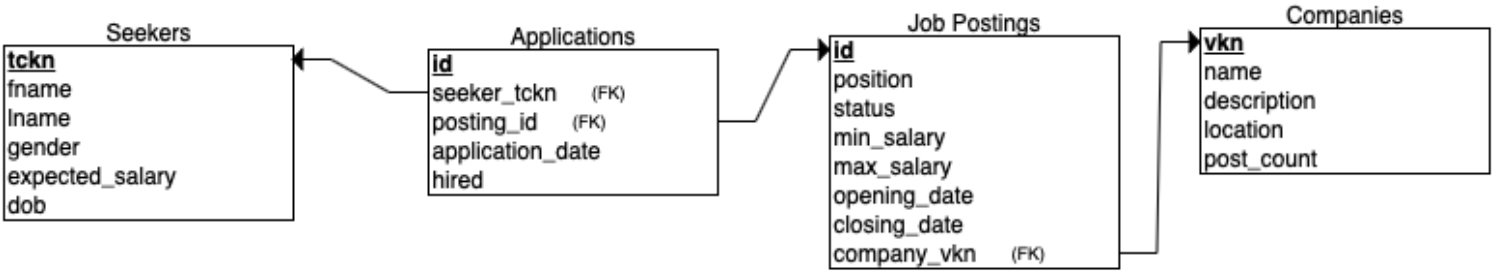
İşçi Bulma Kurumu Bilgi Sistemi

Alp Yüzbaşıoğlu	Izzet Birden	Talha Bacak	Tahir Can Özer
20501032	20501067	16011038	17011061

A. ER Diagram



B. Relational Model Diagram



C. Tablolar

1. Seekers

```
18  SELECT * FROM SEEKERS;
```

Data Output | Explain | Messages | Notifications

	tckn [PK] character varying (12)	fname character varying (50)	lname character varying (50)	gender character (1)	expected_salary integer	dob date
1	13810608984	Aydın	Sancak	E	5184	1983-01-05
2	34552736780	Saffet	Bayrak	E	7464	1987-10-11
3	55447684306	Türkan	Gürsoy	K	5328	1989-10-26
4	84795448964	Canan	Ersoy	K	11404	1995-07-07
5	83752123944	İzzet	Birden	E	24883	1996-04-06
6	68311147708	Can	Arat	E	3628	1999-11-05
7	43013920552	Aydın	Oral	E	25920	1984-12-12
8	40445977628	Kemal	Mutlu	E	10782	1975-06-12
9	68198602000	Zeynep	Yılmaz	K	7257	1988-07-25
10	71080755272	Ali	Yılmaz	E	9331	1990-02-24

2. Applications

```
18  SELECT * FROM APPLICATIONS;
```

Data Output | Explain | Messages | Notifications

	id [PK] integer	seeker_tckn character varying (12)	posting_id integer	application_date date	hired character (1)
1	1	83752123944	2	2019-12-12	1
2	2	96637756762	1	2020-01-03	0
3	3	96637756762	13	2020-01-03	0
4	4	56179376644	7	2019-10-07	1
5	5	38934400696	10	2020-01-01	1
6	6	87344897336	2	2019-12-12	0
7	7	87344897336	9	2019-07-29	1
8	8	40445977628	1	2020-01-03	0
9	9	40445977628	11	2020-01-03	0
10	10	68198602000	4	2020-07-04	0

3. Postings

```
18  SELECT * FROM POSTINGS;
```

Data Output | Explain | Messages | Notifications

	id [PK] integer	position character varying (100)	status character varying (6)	company_vkn character varying (12)	min_salary integer	max_salary integer	opening_date date	closing_date date
1	1	Veritabanı Uzmanı	open	7017438581	5000	10000	2020-01-03	2021-01-01
2	2	İş Analisti	closed	4291657945	7000	10000	2019-10-26	2019-12-12
3	3	Endüstri Mühendisi	open	6680288285	3000	11000	2019-10-11	2021-11-06
4	4	Sistem Analisti	open	9999248558	7000	10000	2019-05-10	2021-02-24
5	5	İş Analisti	open	9999248558	7000	10000	2020-07-04	2021-06-02
6	6	Operasyon Yöneticisi	open	3736925410	7000	12000	2019-02-01	2021-12-08
7	7	Yazılım Mühendisi	closed	3018532388	6500	10000	2019-03-07	2019-10-07
8	8	İş Analisti	open	7017438581	10000	15000	2020-10-05	2021-04-11
9	9	Risk Uzmanı	closed	2414000980	6500	7000	2019-12-26	2019-07-29
10	10	Yazılım Mühendisi	closed	3225281796	12000	15000	2019-11-07	2020-01-01

4. Companies

```
18  SELECT * FROM COMPANIES;
```

Data Output | Explain | Messages | Notifications

	vkn [PK] character varying (12)	name character varying (50)	description character varying (250)	location text[]	post_count integer
1	8049007111	Garanti Teknoloji	Türkiye'nin ilk 3 bankası içine ...	{Istanbul,Iz...	0
2	4710261098	Mercedes Benz	Türkiye'de İstanbul'da ve Aksa...	{Istanbul,Ad...	0
3	3276892154	NCR	Dünyanın en iyi finansal hizme...	{Ankara,Izmi...	0
4	3247717522	Ziraat Bankası	Türkiye'nin son 100 yıldır en iyi...	{Istanbul,Ad...	0
5	7421803526	Hepsiburada	E-ticaret sektörünü bir adım öt...	{Ankara,Izmi...	1
6	3018532388	Turkcell	Türkiye'nin en büyük telekomü...	{Istanbul,Iz...	1
7	2414000980	Akbank	Türkiye'nin en fazla aktif müşt...	{Istanbul,An...	1
8	6680288285	Aselsan	Geliştirdiği savunma teknolojil...	{Istanbul,An...	1
9	4291657945	Kuveyt Türk Katılım Bankası	Katılım bankaları arasında 1., t...	{Istanbul,An...	1
10	1949942318	Tüpraş	Türkiye'nin en çok kar eden şir...	{Ankara,Ada...	1

D. Maddeleri Sağlayan Kod Blokları

2. Tablolarınızda primary key ve foreign key kısıtlarını kullanmalısınız.

```
CREATE TABLE IF NOT EXISTS APPLICATIONS (
    id INT PRIMARY KEY,
    seeker_tckn VARCHAR(12) NOT NULL,
    posting_id INT NOT NULL,
    application_date DATE DEFAULT CURRENT_DATE NOT NULL,
    hired CHAR DEFAULT 0 NOT NULL,
    CONSTRAINT fk_seeker
        FOREIGN KEY(seeker_tckn)
```

```
        REFERENCES SEEKERS(tckn)
        ON DELETE CASCADE,
    CONSTRAINT fk_posting
        FOREIGN KEY(posting_id)
        REFERENCES POSTINGS(id)
        ON DELETE NO ACTION
);
```

3. En az bir tabloda silme kısıtı ve sayı kısıtı olmalıdır.

```
CREATE TABLE IF NOT EXISTS POSTINGS (
    id INT PRIMARY KEY,
    position VARCHAR(100),
    status VARCHAR(6) DEFAULT 'open' NOT NULL,
    company_vkn VARCHAR(12) NOT NULL,
    min_salary INT NOT NULL CHECK (min_salary > 0),
    max_salary INT NOT NULL,
    opening_date DATE,
    closing_date DATE,
    FOREIGN KEY(company_vkn) REFERENCES COMPANIES(vkn)
);
```

4. Arayüzden en az birer tane insert, update ve delete işlemi gerçekleştirilebilmelidir.

```
INSERT INTO SEEKERS VALUES
('79266970006', 'Can', 'Eryürür', 'E', 2400, '12-02-1983');
UPDATE SEEKERS SET dob = '03-03-1993' WHERE tckn = '79266970006';
DELETE FROM SEEKERS WHERE tckn = '79266970006';
```

5. Arayüzden girilecek bir değere göre ekrana sonuçların listelendiği bir sorgu yazmalısınız.

```
SELECT * FROM SEEKERS_AGE WHERE AGE < 30;
```

6. Arayüzden çağrılan sorgulardan en az biri “view” olarak tanımlanmış olmalıdır.

```
CREATE OR REPLACE VIEW SEEKERS_AGE AS
SELECT tckn,
        CONCAT(fname, ' ', lname) AS name,
        expected_salary,
        CAST(DATE_PART('YEAR', AGE(dob)) AS INT) AS age
FROM SEEKERS;
```

7. En az bir adet “sequence” oluşturmalı ve arayüzden yapılacak insert sırasında ilgili sütundaki değerlerin otomatik olarak atanmasını sağlamalısınız.

```
CREATE SEQUENCE IF NOT EXISTS SEQ_APPLICATION_ID
START WITH 1 INCREMENT BY 1 NO CYCLE;


INSERT INTO APPLICATIONS VALUES
(nextval('SEQ_APPLICATION_ID'), '83752123944', 2, '12-12-2019', 1);
```

8. Arayüzden çağrılan sorgulardan en az birinde union veya intersect veya except kullanmış olmalısınız.

```
SELECT * FROM SEEKERS_AGE WHERE AGE > 30 AND expected_salary < 12500
INTERSECT
SELECT * FROM SEEKERS_AGE WHERE tckn IN (SELECT seeker_tckn FROM APPLICATIONS);
```

9. Sorgularınızın en az biri aggregate fonksiyonlar içermeli, having ifadesi kullanılmalıdır.

```
SELECT tbl2.name,
       tbl1.posting_count
FROM (
    SELECT company_vkn,
           COUNT(*) AS posting_count
    FROM POSTINGS
    GROUP BY company_vkn
    HAVING COUNT(*) > 1
) tbl1
INNER JOIN COMPANIES tbl2
ON tbl1.company_vkn = tbl2.vkn;
```

10. Arayüzden girilen değerleri parametre olarak alıp ekrana sonuç döndüren 3 farklı SQL fonksiyonu tanımlamış olmalısınız. Bu fonksiyonlarda en az birer tane “record” ve “cursor” tanımı-kullanımı olmalıdır.

```
CREATE OR REPLACE FUNCTION raise_expected_salary(percentage INT)
RETURNS TABLE (
    fname VARCHAR(50),
    lname VARCHAR(50),
    raised_expected_salary INT,
    dob DATE
)
AS $$
BEGIN
    IF (percentage > 0) THEN
        --First update table
        UPDATE SEEKERS SET expected_salary = expected_salary * (100 +
percentage) / 100;
        --Then return it
        RETURN QUERY
            SELECT
                tbl1.fname,
                tbl1.lname,
                tbl1.expected_salary AS raised_expected_salary,
                tbl1.dob
            FROM
                SEEKERS tbl1;
    ELSE
        RAISE EXCEPTION 'Percantage must be higher than 0';
    END IF;
END;
$$ LANGUAGE PLPGSQL;
```

```
CREATE OR REPLACE FUNCTION find_companies(count INT)
RETURNS SETOF VARCHAR
AS $$
BEGIN
    IF (count IS NOT NULL) THEN
        RETURN QUERY
            SELECT name
            FROM COMPANIES
            WHERE array_length(location, 1) > count;
    ELSE
        RAISE EXCEPTION 'city cant be null';
    END IF;
END;
$$ LANGUAGE PLPGSQL;
```

```
CREATE OR REPLACE FUNCTION find_max_salary_by_position(job_position VARCHAR, OUT
max_salary INT)
AS $$
BEGIN
    IF (job_position IS NOT NULL) THEN
            SELECT MAX(tbl1.max_salary) INTO max_salary
            FROM POSTINGS tbl1
            WHERE upper(tbl1.position) = upper(job_position);
    ELSE
        RAISE EXCEPTION 'position cant be null';
    END IF;
END;
$$ LANGUAGE PLPGSQL;
```

```
CREATE FUNCTION AVGSALARY (SEX CHAR(1)) RETURNS NUMERIC AS $$
DECLARE
sum_sal NUMERIC;
i integer;
curs CURSOR FOR SELECT expected_salary FROM seekers WHERE gender=sex;

BEGIN
sum_sal := 0;
i =0;
FOR satir IN curs LOOP
sum_sal := sum_sal + satir.expected_salary;
i := i + 1;
END LOOP;

RETURN sum_sal/i;
END;
$$ LANGUAGE PLPGSQL;
```

11. 2 adet trigger tanımlamalı ve arayüzden girilecek değerlerle tetiklemelisiniz. Trigger'ın çalıştığına dair arayüze bilgilendirme mesajı döndürülmelidir.

```
CREATE OR REPLACE FUNCTION CHECK_AGE() RETURNS trigger AS $CHECK_AGE$
    BEGIN
        RAISE INFO 'CHECK_AGE function is triggered';
        IF CAST(DATE_PART('YEAR', AGE(NEW.dob)) AS INT) < 18 THEN
            RAISE EXCEPTION 'Children under the age of 18 cannot register to the
system';
        END IF;

        RETURN NEW;
    END;
$CHECK_AGE$ LANGUAGE plpgsql;

CREATE TRIGGER CHECK_AGE BEFORE INSERT OR UPDATE ON SEEKERS
    FOR EACH ROW EXECUTE PROCEDURE CHECK_AGE();
```

```
CREATE OR REPLACE FUNCTION CALC_POST_COUNT() RETURNS TRIGGER AS $POST_COUNT$
    BEGIN
        RAISE INFO 'calc_post_count function is triggered';
        IF (TG_OP = 'DELETE') THEN
            UPDATE COMPANIES SET post_count = post_count -1 WHERE vkn =
OLD.company_vkn;
        ELSIF (TG_OP = 'UPDATE') THEN
            UPDATE COMPANIES SET post_count = post_count -1 WHERE vkn =
OLD.company_vkn;
            UPDATE COMPANIES SET post_count = post_count +1 WHERE vkn =
NEW.company_vkn;
        ELSIF (TG_OP = 'INSERT') THEN
            UPDATE COMPANIES SET post_count = post_count +1 WHERE vkn =
NEW.company_vkn;
        END IF;
        RETURN NULL;
    END;
$POST_COUNT$ LANGUAGE plpgsql;

CREATE TRIGGER POST_COUNT
AFTER INSERT OR UPDATE OR DELETE ON POSTINGS
    FOR EACH ROW EXECUTE PROCEDURE CALC_POST_COUNT();
```