MALAVIYA NATIONAL INSTITUTE OF TECHNOLOGY JAIPUR

Lab Sheet : JOINED RELATIONS

GOAL:

- *What are joined Relations?*
  - *Brief overview of joined relations*
- *Getting familiar with different join operations*
  - *Inner join*
  - *Left outer join*
  - *Natural inner join*
  - *Right outer join*
  - *Full outer join*

WHAT ARE JOINED RELATIONS:

Overview of joined relations:

SQL provides not only the basic Cartesian-product mechanism for joining tuples of relations found in its earlier versions, but, SQL also provides various other mechanism of joining relations including condition joins and natural joins. These additional operations are typically used as subquery expressions in the **FROM** clause.

Each of the variants of the join operations in SQL consists of a **join type and a join condition**. The join condition defines which tuples in the two relations match and what attributes are present in the result of the join. The join type defines how tuples in each relation that do not match any tuple in the other relation (based on the join condition).

Following tables shows **join type and join condition:**

Join Type
Inner join
Left Outer Join
Right Outer Join
Full Outer Join

Join Condition
natural **on** <predicate> **using** <A_1,A_2,$A_{3...}$,A_n>

To understand how join works, Let's create following tables :

```
Create table loan (
        loanNo  varchar(10),
        branch_name   varchar(20),
        amount  int(5),
        constraint  loan_pk primary key (loanNo)
    );
```

```
Create table borrower (
        customer_name varchar(20),
        loanNo  varchar(10),
        constraint  borrower_pk  primary key (loanNo)
);

Create table depositor (
        customer_name   varchar(20),
        account_number  varchar(20),
        constraint depositor_pk primary key (customer_name)
);
```

Now populate the tables with following data:

LOAN

loanNo	branch_Name	amount
L-170	Downtown	3000
L-230	Redwood	4000
L-260	Perryridge	1700
L-155	Washigton	4500

BORROWER

Customer_name	loanNo
Jones	L-170
Smith	L-230
Hayes	L-155

DEPOSITOR

Customer_name	Account_number
Jones	HDBC2044
Smith	HDBC2055
Hayes	HDBC2066
Chris	HDBC2077
Adam	HDBC2088

Getting Familiar with Different Join Operations:

- **Inner Joins:**

We illustrate the use of inner joins through an example:

Select * from (loan l **inner join** borrower b on l.loanno =b.loanno);

//observe the output carefully specifically field names and values of resultant.

The expression computes the theta join of the loan and the borrower relations, with the join condition being loan.loan-number = borrower.loan-number. The attributes of the result consist of the attributes of the left-hand-side relation followed by the attribute of the right-hand-side relation.

- **Left outer joins:**

Illustrates the use of left outer joins through an example:

Select * **from** (loan l **left outer join** borrower b on l.loanno = b.loanno);

//observer the output carefully

We can compute the left outer join operation logically as follows.

First, compute the result of the inner join as before. Then, for every tuple t in the left-hand-side relation loan that does not match any tuple in the right-hand-side relation borrower in the inner join, add a tuple r to the result of the join: The attributes of tuple r that are derived from the left-hand-side relation are filled in with the values from tuple t,and the remaining attributes of r are filled with null values.

- **Natural inner joins:**

Consider an example of the natural join operation:

select * from (loan **natural inner join** borrower);

//observer the output carefully and compare with previous outputs

This expression computes the natural join of the two relations. The only attribute name common to loan and borrower is loan-number.

Difference between natural join and inner join is that the attribute loan-number appears only once in the result of the natural join but it comes twice in case of inner join.

- **Right outer join:**

Consider an example of the right outer join operation:

select * from (loan l **right outer join** borrower b on l.loanno =b .loanno);

//observer the output carefully and compare with left-outer join

The right outer join is symmetric to the left outer join. Tuples from the right-hand-side relation that do not match any tuple in the left-hand-side relation are padded with nulls and are added to the result of the right outer join.

- **Full outer join**

Consider an example of the full outer join operation:

SELECT * from (loan l **full outer join** borrower b on l.loanno=b.loanno);

//observer the output carefully and compare with previous outputs

The full outer join is a combination of the left and right outer-join types. After the operation computes result of the inner join, it extends with nulls tuples from the left-hand-side relation that did not match with any from the right-hand-side, and adds them to the result.

Similarly, it extends with nulls tuples from the right-hand- side relation that did not match with any tuples from the left-hand-side relation and adds them to the result.

- **How to use various above joined relations in SQL statements:**

Example: Find all customers who have an account but no loan at the bank?

```
SELECT depositor.customer-name
    FROM  (depositor left outer join borrower
        on depositor.customer-name = borrower.customer-name)
    WHERE borrower.customer-name  is null;
```

Example: Find all customers who have either an account or a loan (but not both) at the bank?

```
SELECT customer-name
    FROM (depositor natural full outer join borrower)
        WHERE account-number is null OR loan-number is null;
```

To do Exercise:

i. **Determine the names of sailors who are older than the oldest sailor with a rating of 10?**

ii. **Determine the names of sailors who have reserved all boats?**

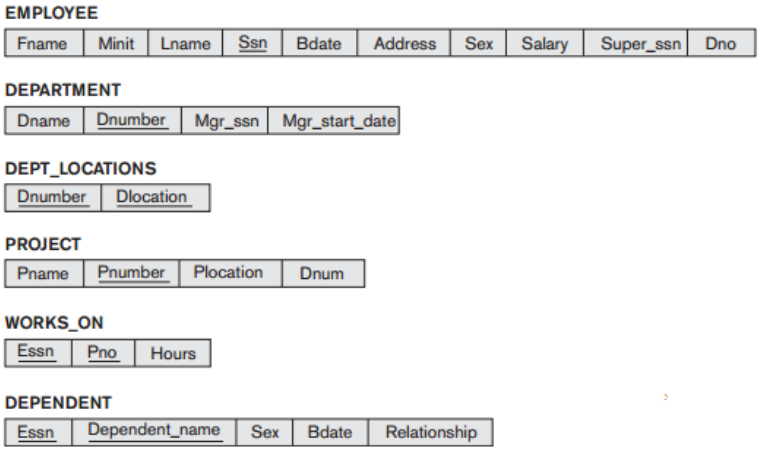
iii. **Determine the average age of sailors with a rating of 10?**

iv. **Determine the name and the age of the oldest sailor?**

v. **Count the number of sailors?**

vi. **For each red boat, find the number of reservations for this boat?**

Exercise: a) Create a database for an organization as given diagram of relational database schema.



Relation Name	Attribute Name	Data Type	Constraint
department	dnumber	Integer	Primary key
	dname	varchar	Not null
	MGRssn	integer	Not null
	MGRstartdate	integer	Not null
dependent	Essn	integer	Primary key, Foreign Key
	dependent_name	varchar	Primary key
	Sex	varchar	Not null
	Bdate	varchar	Not null
	relationship	varchar	Not null
dep_location	Dnumber	integer	Primary key, Foreign key
	Dlocation	varchar	Primary key
employee	first_name	varchar	Not Null
	MINIT	varchar	Not Null
	last_name	varchar	Not null
	Ssn	integer	Primary key
	Bdate	varchar	Not Null
	Address	varchar	Not Null
	Sex	varchar	Not Null
	Salary	integer	Not Null
	SUPERssn	integer	Null
	Dno	integer	Not Null, Foreign key
project	Pname	varchr	Not null
	Pnumber	integer	Primary key
	Plocation	varchar	Not null
	Dnum	integer	Not null
works_on	Essn	integer	Primary key, Foreign key
	Pno	integer	Primary key, Foreign key
	Hours	varchar	null

b) Populate the Database with the below dataset in each table:

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse