ERASMUS+

Enriching lives, opening minds

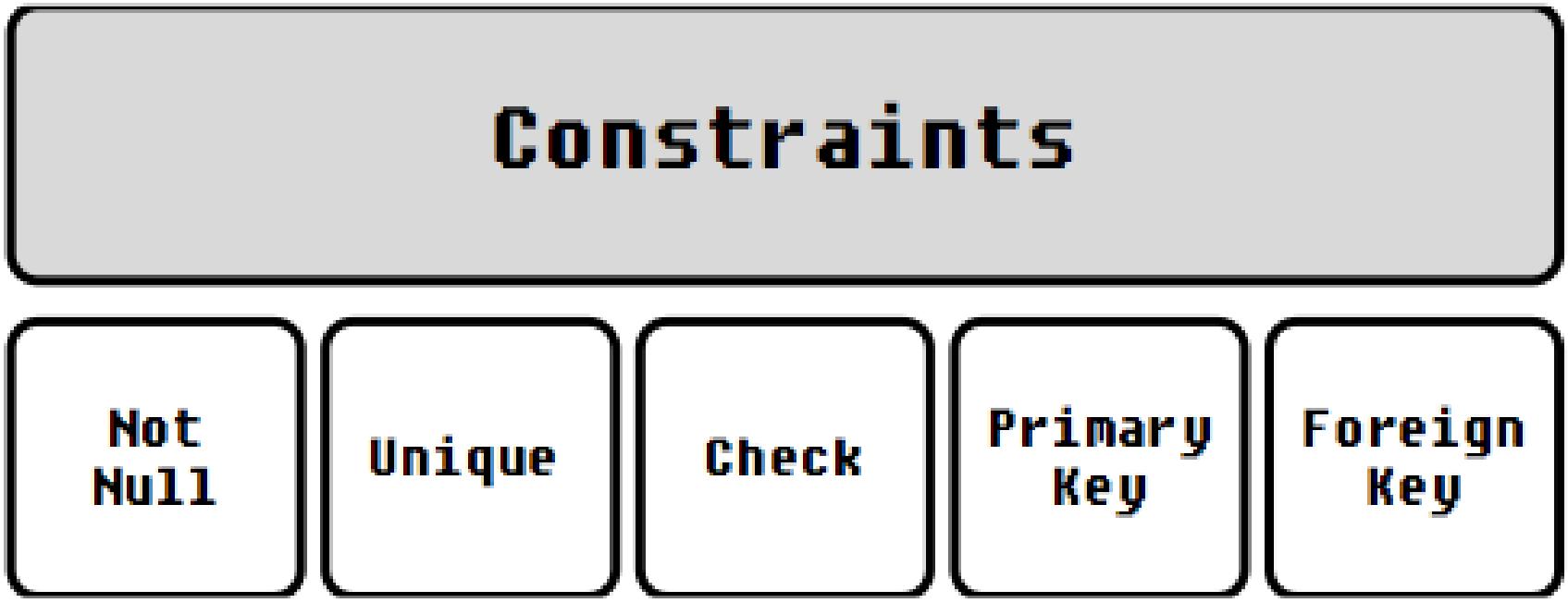
ADO.NET: SQL

ИНСТИТУТ
БИЗНЕСА
БГУ

```
INSERT INTO
tP (FirstName, LastName, Age, NetWorth)
VALUES
('Andrey',
'Melnichenko','$9.1','43');
UPDATE tP SET Age='47'
WHERE (LastName='Melnichenko');
DELETE FROM tP WHERE Age='47';
```

```
CREATE DATABASE db;
CREATE TABLE [tW] (
    [Id]        BIGINT          IDENTITY (1, 1)
NOT NULL,
    [FirstName] NVARCHAR (MAX) NULL,
    [LastName] NVARCHAR(MAX) NULL,
    [HID] BIGINT NULL,
    PRIMARY KEY CLUSTERED ([Id] ASC)
);
```

SQL Constraints



- **NOT NULL** – Определяет, что поле не должно содержать значения NULL

- **UNIQUE** – определяет, что каждый ряд для данного поля должен иметь уникальное значение
- **PRIMARY KEY** = NOT NULL + UNIQUE. Гарантирует, что поле имеет уникальную идентификацию, что помогает найти нужную запись более просто и быстро.
- **FOREIGN KEY** – связь данного поля этой таблицы с полями другой таблицы.
- **CHECK** – Гарантирует, что поле удовлетворяет определенным условиям
- **DEFAULT** – Определяет значение по умолчанию для поля.

FOREIGN KEY

FOREIGN KEY в таблице указывает на PRIMARY KEY в другой таблице.

tP

ID	First Name	Last Name	Age	Net-worth
1	**Vladimir**	**Potanin**	55	15,4
2	Mihail	**Fridman**	51	14,6
3	Alisher	**Usmanov**	62	14,4

tW

ID	First Name	Last Name	HID
1	Nathalia	Potanina	1
2	Olga	Fridman	2
3	Irina	Winner	3

Если мы хотим обозначить тот факт, что поле **tW.HID** указывает на поле **tP.ID** то FOREIGN KEY.
Такое ограничение позволит предотвратить действия, которые могут повредить связь между таблицами.
Также предотвращается вставка неправильных данных в FOREIGN KEY.

```
CREATE TABLE [tW] (
 [Id] BIGINT IDENTITY (1, 1) NOT NULL,
 [FirstName] NVARCHAR (MAX) NULL,
 [LastName] NVARCHAR(MAX) NULL,
 [HID] INT NOT NULL FOREIGN KEY REFERENCES
[tP](Id) );
```

CHECK

Ограничение CHECK используется чтобы ограничить диапазон значений, которые могут быть размещены в данном поле.

```
CREATE TABLE [tW3] (
[Id]  BIGINT IDENTITY (1, 1) NOT NULL,
    [FirstName] NVARCHAR (MAX) NULL,
    [LastName] NVARCHAR(MAX) NULL,
    [HID] INT NOT NULL CHECK (HID>0) FOREIGN
KEY REFERENCES [Table](Id) );
```

Можно сделать и так

```
CREATE TABLE [tW4] (
    [Id] BIGINT IDENTITY (1, 1) NOT NULL,
    [FirstName] NVARCHAR (MAX) NULL,
    [LastName] NVARCHAR(MAX) NULL,
    [HID] INT NOT NULL
        FOREIGN KEY REFERENCES [Table](Id),
  CONSTRAINT chk_Person CHECK (HId>0 AND
FirstName='Inna')
);
```

```
ALTER TABLE tW4
DROP CONSTRAINT [chk_Person]

ALTER TABLE tW4
ADD CONSTRAINT chk_Person CHECK (HId>0 AND
FirstName='Inna')
```

CREATE INDEX

SELECT LastName FROM tP WHERE FirstName = 'Vladimir';

В такой конфигурации необходимо осуществить полный перебор записей и найти все,
где

FirstName = 'Vladimir'

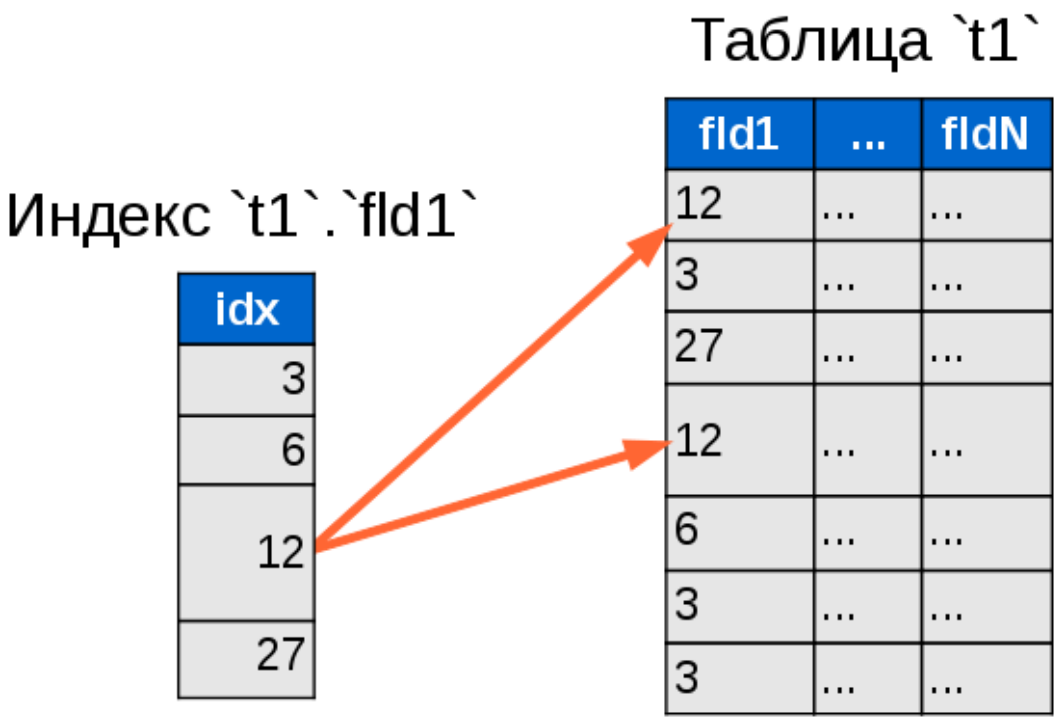
При использовании индекса по полю FirstName создается бинарное дерево, и поиск, естественно, идет значительно быстрее. Но на создание бинарного дерева и включение в него еще одного элемента надо время.

Чтобы создать индекс на поле FirstName надо выполнить нижеследующий SQL код.

```
CREATE INDEX PInd ON tP (FirstName)
```

Можно создать индекс на комбинацию 2-х полей

CREATE INDEX PInd
ON tP (LastName,
FirstName)



CREATE VIEW

```
CREATE VIEW [ALL] AS SELECT * FROM tW


CREATE TABLE [tW7] (
    [Id] BIGINT IDENTITY (1, 1) NOT NULL,
    [FirstName] NVARCHAR (MAX) NULL,
    [LastName] NVARCHAR(MAX) NULL,
```

```
    [HID] INT NOT NULL FOREIGN KEY REFERENCES
[Table](Id),
  CONSTRAINT chk_P CHECK (HId>0)
);
INSERT INTO tW7 ([FirstName], [LastName],
[HID]) VALUES ('2','2','1');
```