Android

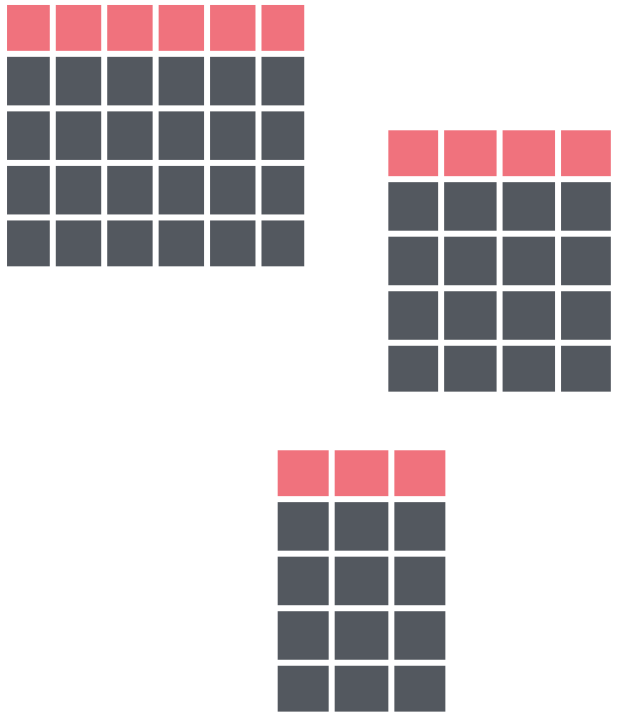
Author : Do Phu Quy
Email : phuquycntt@gmail.com
Phone : 0935 366 007

Realm

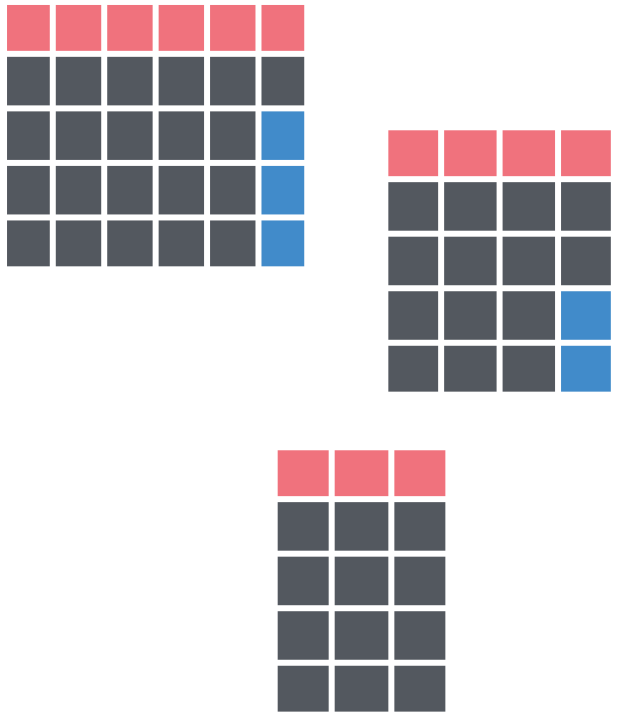
Building a mobile database

Why a new database?

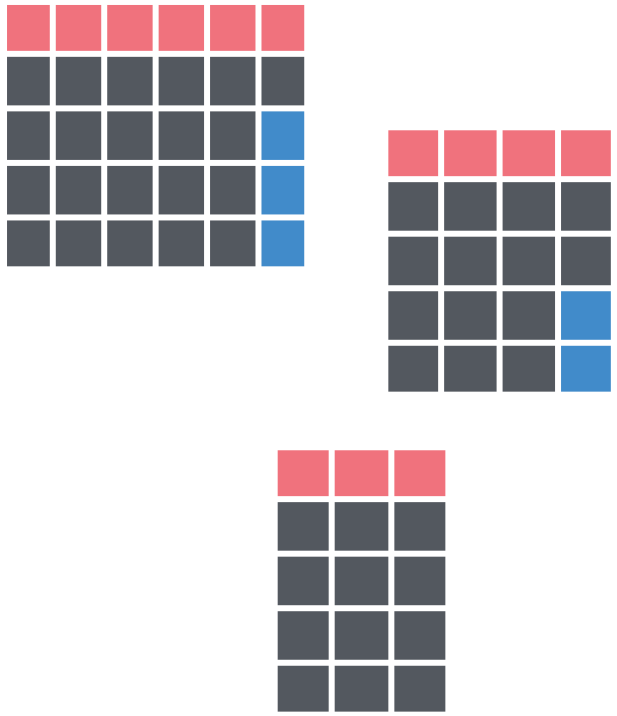
Once upon a time



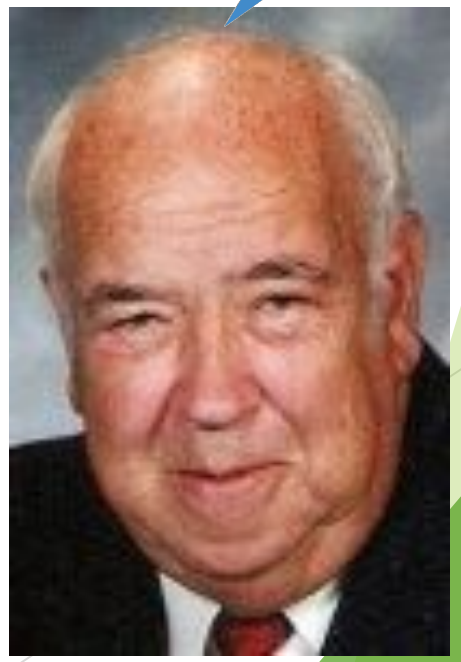
Once upon a time

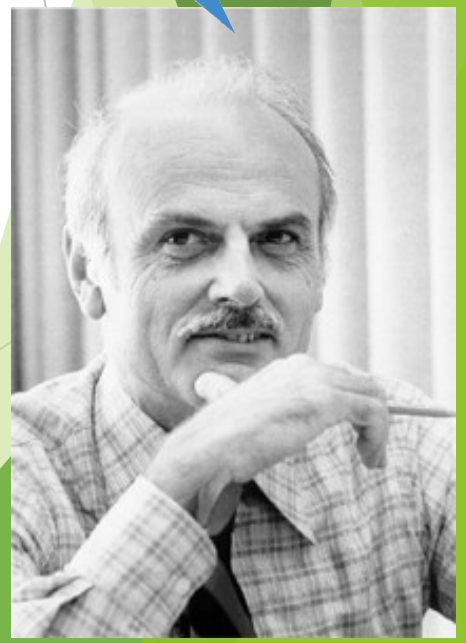


Once upon a time

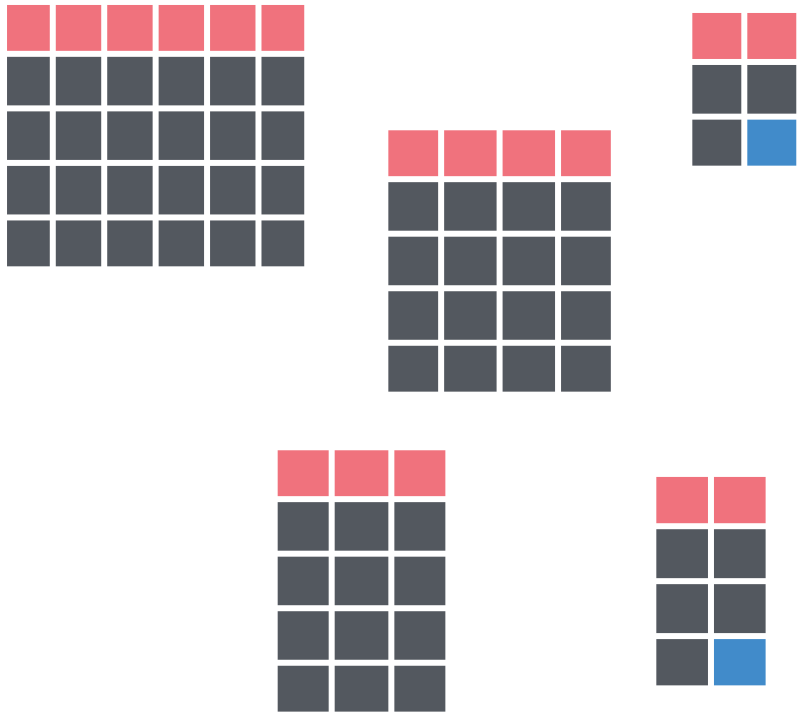


BCN
F

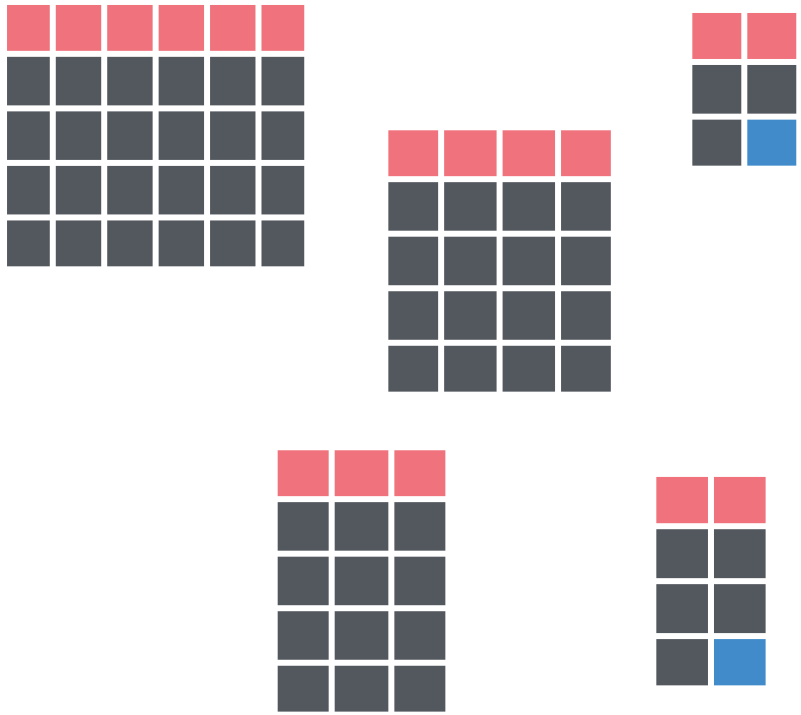




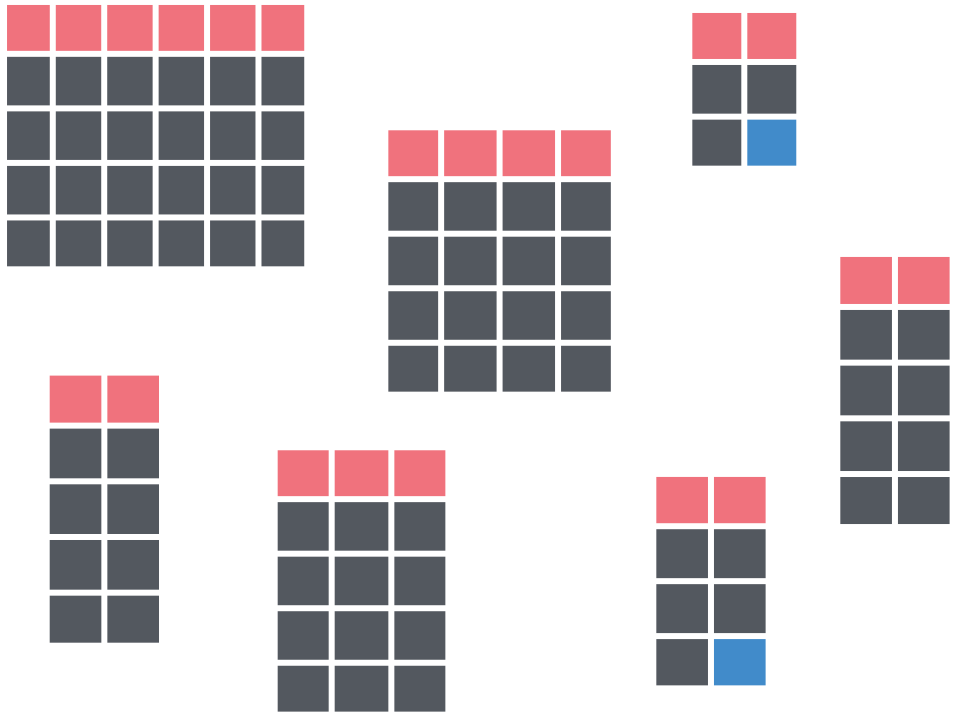
Once upon a time



Once upon a time



Once upon a time



Once upon a time

```
SELECT   owner.name,      dog.name,      city.name
  FROM   owner
         INNER JOIN dog ON owner.dog_id = dog.id
         INNER JOIN city ON owner.city_id = city.id
         WHERE owner.name = 'Frank'
```

Once upon a time

```
String query = "SELECT " + Owner.NAME + ", " + Dog.NAME + ", " + City.NAME
        + " FROM " + Owner.TABLE_NAME
        + " INNER JOIN " + Dog.TABLE_NAME + " ON " + Owner.DOG_ID + " = " + Dog.ID
        + " INNER JOIN " + City.TABLE_NAME + " ON " + Owner.CITY_ID + " = " + City.ID
        + " WHERE " + Owner.NAME = "'" + escape(queryName) + "'";
```

Abstract the problem away

“All non-trivial abstractions, to some degree, are leaky.”

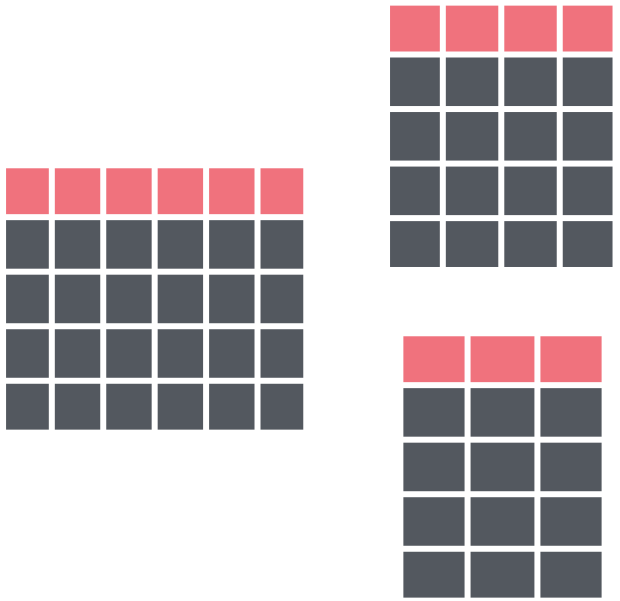
Why a new database?

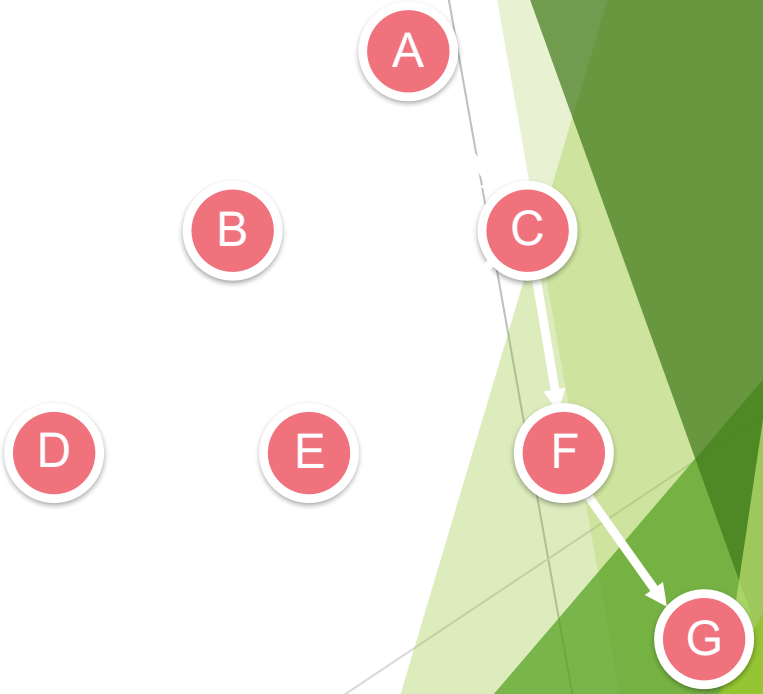
What is **Realm**?

Zero-copy object store

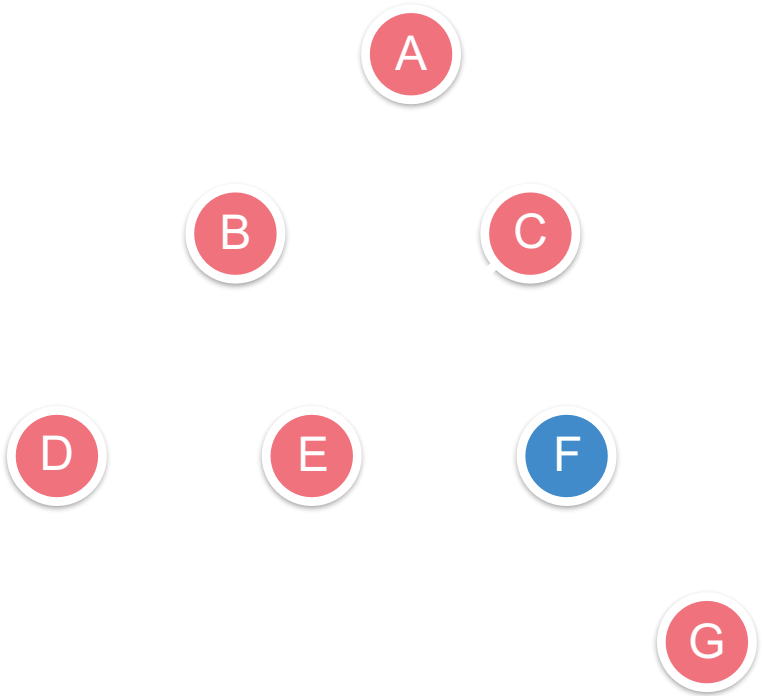
Designed for mobile devices

Object Store

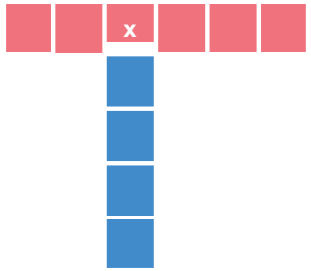


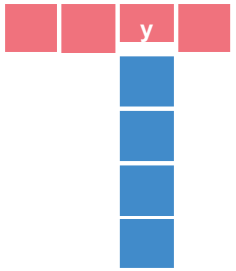


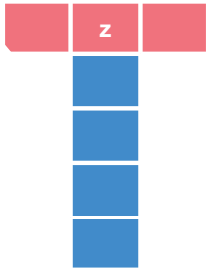
Object Store references

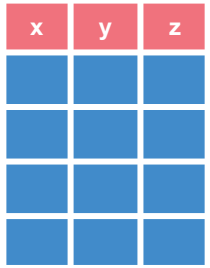


References in SQL









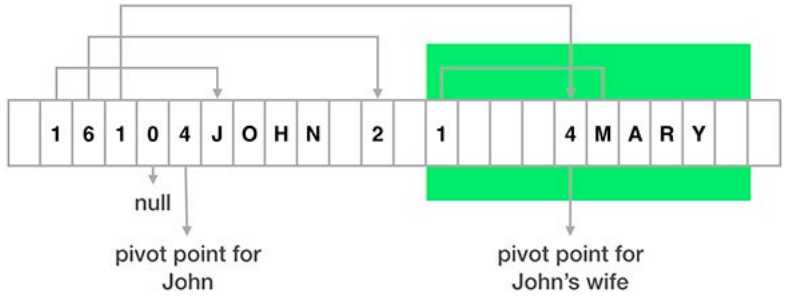

```
SELECT table1.x, table2.y, table3.z
FROM table1
INNER JOIN table2 ON table1.table2_id = table1.id
INNER JOIN table3 ON table1.table3_id = table3.id
```

What is zero-copy?

```
public class MyCursorAdapter extends CursorAdapter {

    // ...

    @Override
    public void bindView(View view, Context context, Cursor cursor) {
        ViewHolder holder = (ViewHolder) view.getTag();
        String foo = cursor.getString(cursor.getColumnIndexOrThrow("foo")); int bar =
        cursor.getInt(cursor.getColumnIndexOrThrow("bar"));
        holder.fooView.setText(foo); holder.barView.setText(Integer.toString(bar));
    }
}
```

What is zero-copy?

- CursorAdapter



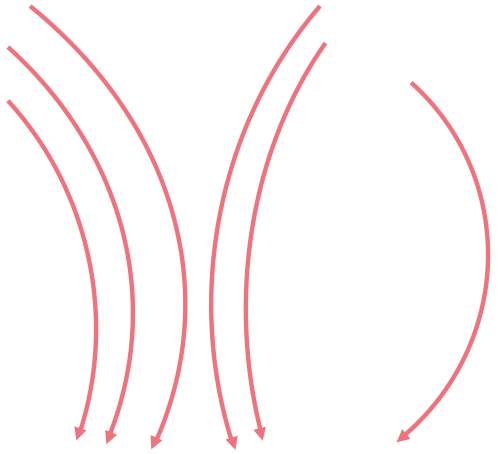

```
public long id() {
   int o = __offset(4);
   return o != 0 ? bb.getLong(o + bb_pos) : 0;
}

public String name() {
   int o = __offset(6);
   return o != 0 ? __string(o + bb_pos) : null;
}
```

```
ByteBuffer bb = ByteBuffer.wrap(bytes); MyObject
obj = MyObject.getRootAsMyObject(bb); long id =
obj.id();
String name = obj.name();
```

Zero-copy in Realm

```
Person {
 • name = Tommy
 • age = 8
 • dog = {
     • name = Lassie
   }
}
```

```
Person {
 • name = Tommy
 • age = 8
 • dog = {
     • name = Lassie
   }
}
```



Realm

Realm

```
Person {
 • name = Tommy
 • age = 8
 • dog = {
     • name = Lassie
   }
}
```

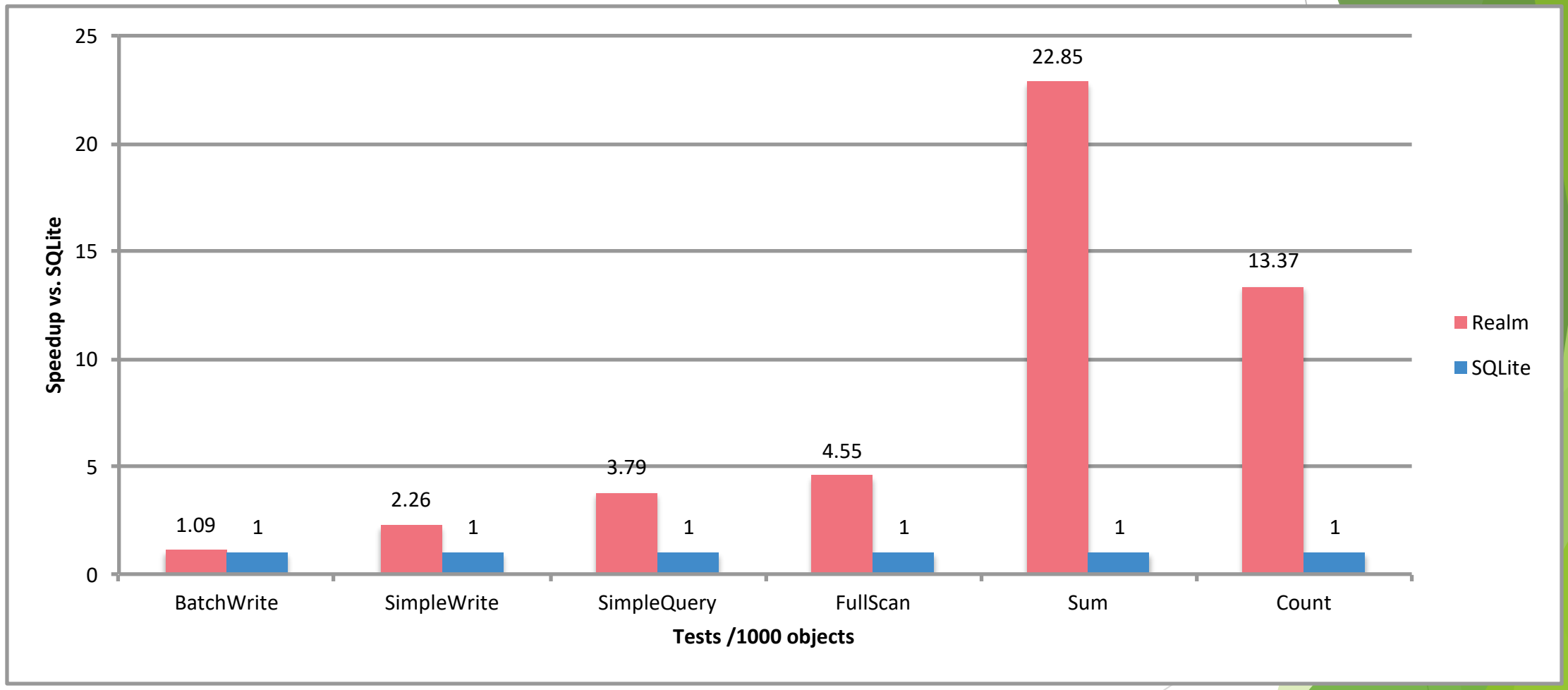
Zero-copy in Realm

```
// Objects
Person javaPerson = new Person(); // Standard Java POJO

Person proxyPerson = realm.copyToRealm(javaPerson); // Convert POJO to Proxy  Person

proxyPerson = realm.createObject(Person.class); // Proxy


// Query results are lazy
RealmResults<Person> queryResults = realm.allObjects(Person.class);

queryResults.get(0) != queryResults.get(0); // Different Java objects

queryResults.get(0).equals(queryResults.get(0)); // Same data
```

ad

rites

Benchmarks



http://static.realm.io/downloads/java/android-benchmark.zip

Implementing Realm Android

Architecture

Realm Object Store API

Realm Internal API

JNI

Realm Core

Zero copy

```
public class Pojo {

	private String foo;
	private int bar;

	public Pojo() {
	}

	public String getFoo() {
		return foo;
	}

	public void setFoo(String foo) {
		this.foo = foo;
	}

	public int getBar() {
		return bar;
	}

	public void setBar(int bar) {
		this.bar = bar;
	}
}
```

```java
public class Pojo extends RealmObject {

    private String foo;
    private int bar;

    public Pojo(String foo, int bar) {
        this.foo = foo;
        this.bar = bar;
    }

    public String getFoo() {
        return foo;
    }

    public void setFoo(String foo) {
        this.foo = foo;
    }

    public int getBar() {
        return bar;
    }

    public void setBar(int bar) {
        this.bar = bar;
    }
}
```

=

```java
@RealmClass
public abstract class RealmObject {

    protected Row row;
    protected Realm realm;

    // ...
}
```

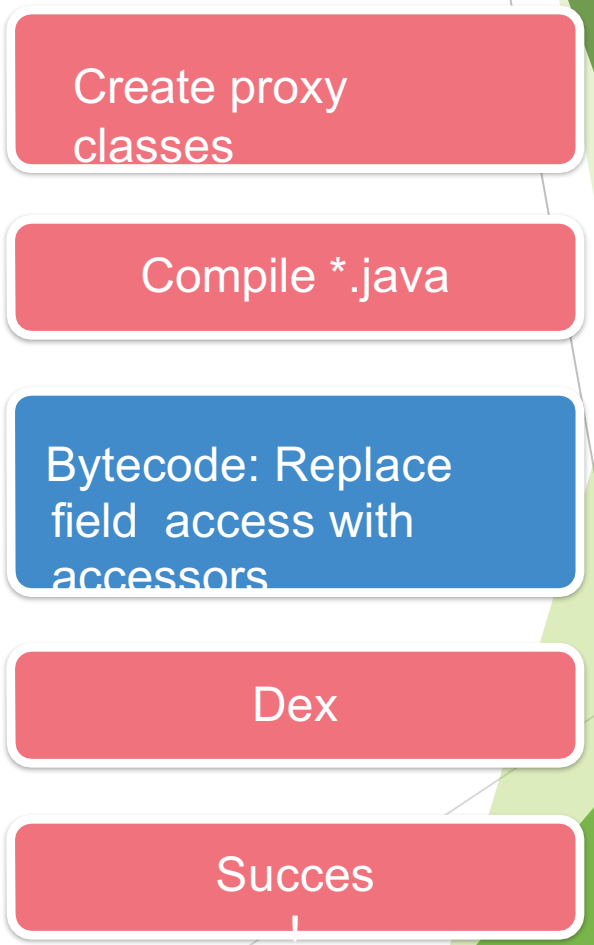
Proxy classes

```
public class PojoRealmProxy extends Pojo
    implements RealmObjectProxy {

    private static long INDEX_FOO;
    private static long INDEX_BAR;

    @Override
    public String getFoo() {
        realm.checkIfValid();
        return (java.lang.String) row.getString(INDEX_FOO);
    }

    @Override
    public void setFoo(String value) {
        realm.checkIfValid();
        row.setString(INDEX_FOO, (String) value);
    }

    @Override
    public void setBar(int value) {
        realm.checkIfValid();
        row.setLong(INDEX_BAR, (long) value);
    }

    @Override
    public String toString() {
        // ..
    }

    @Override
    public int hashCode() {
        // ...
    }

    @Override
    public boolean equals(Object o) {
        // ...
    }

    // Static helper methods ...
}
```

Standard Java objects

```
public class NormalObject extends RealmObject {

      public String foo;  public int bar;

     @Override
      public String toString() {
            return "[" + foo + ";" + bar + "]";
      }
}

NormalObject obj = new NormalObject();  obj.foo =
"DroidCon";
```

Byte code manipulation



Byte code manipulation

```
public class Pojo extends RealmObject {

    public String foo;
    public int bar;

    @Override
    public String toString() {
        return "[" + foo + ";" + bar + "]";
    }

    public String realm$$getFoo() {
        return foo;
    }

    public void realm$$setFoo(String s) {  foo
        = s;
    }

    public int realm$$getBar() {
        return bar;
    }

    public int realm$$setBar(int i) {
        return bar = i;
    }
}
```

```
public class Pojo extends RealmObject {

    public String foo;
    public int bar;

    @Override
    public String toString() {
        return "[" + realm$$getFoo() + ";" + realm$$getBar() + "]";
    }

    public String realm$$getFoo() {
        return foo;
    }

    public void realm$$setFoo(String s) {  foo
        = s;
    }

    public int realm$$getBar() {
        return bar;
    }

    public int realm$$setBar(int i) {
        return bar = i;
    }
}
```

```
Pojo pojo = new Pojo();
pojo.realm$$setFoo("DroidCon");
```

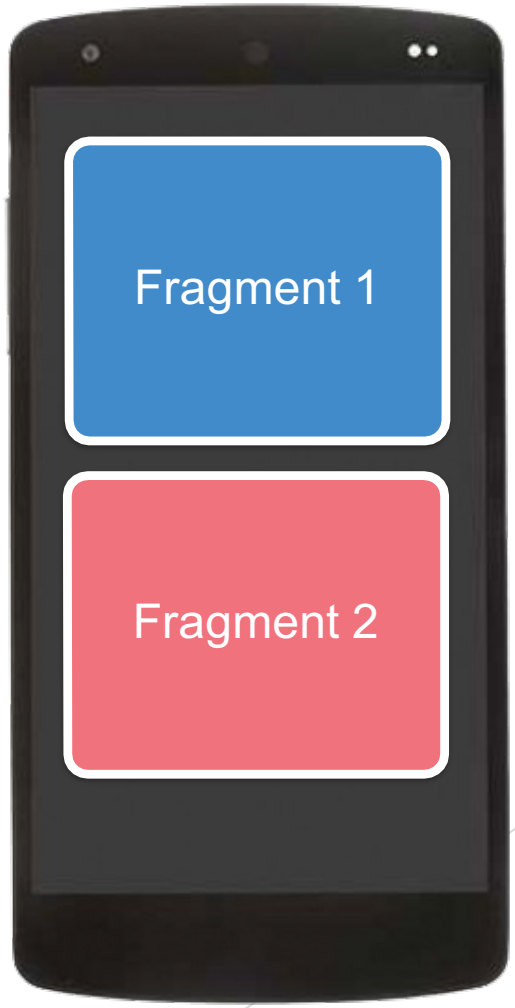
Method count

Picasso: ~600
GSON: ~950
OrmLite: ~2400
RxJava: ~3500
Support library: ~12300

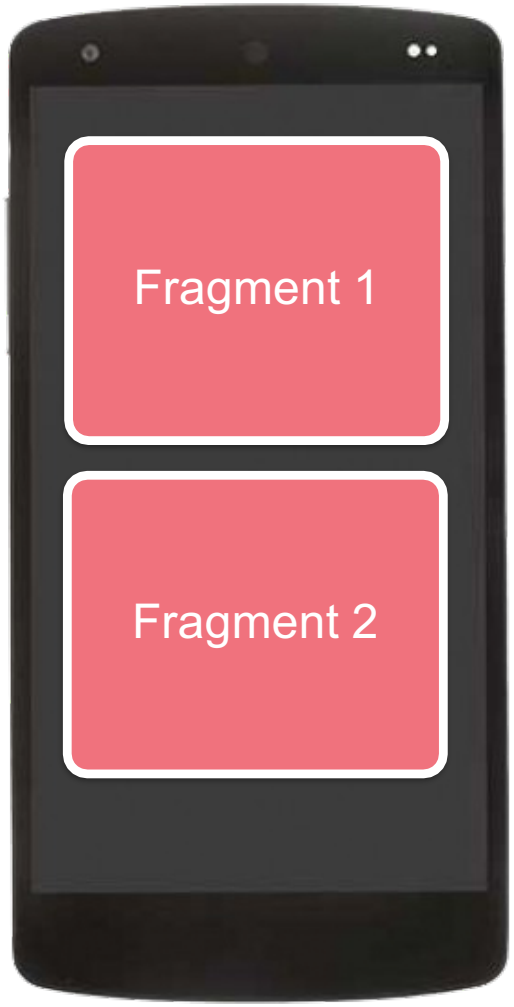
What does consistency mean?

- UI consistency: Visible data should represent one valid state.


Fragment 1
Fragment 2


Fragment 1
Fragment 2

Consistency today



Consistency today

ContentProvider

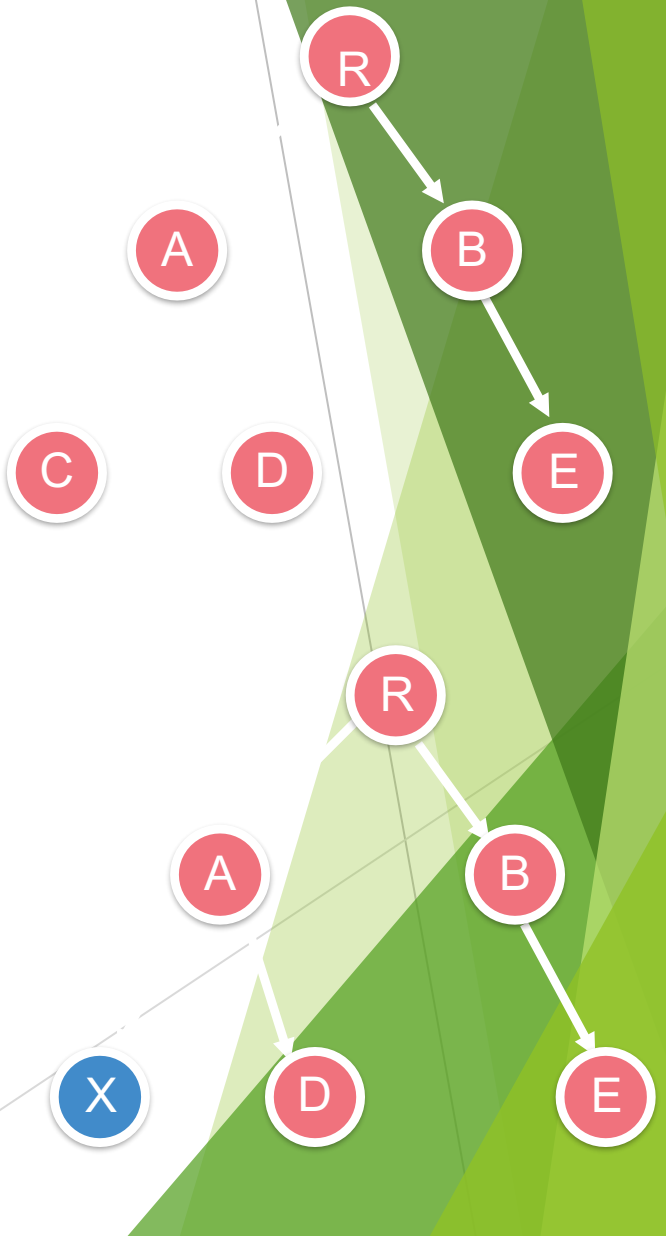
MVCC

Multiversion concurrency control

Multiple read transactions

Data is versioned

One write transaction



Consistency in Realm

UI thread

Background thread

```
queue.next();
```

```
thread.start();
```

```
queue.next();
```

Consistency in Realm

UI thread

Background thread

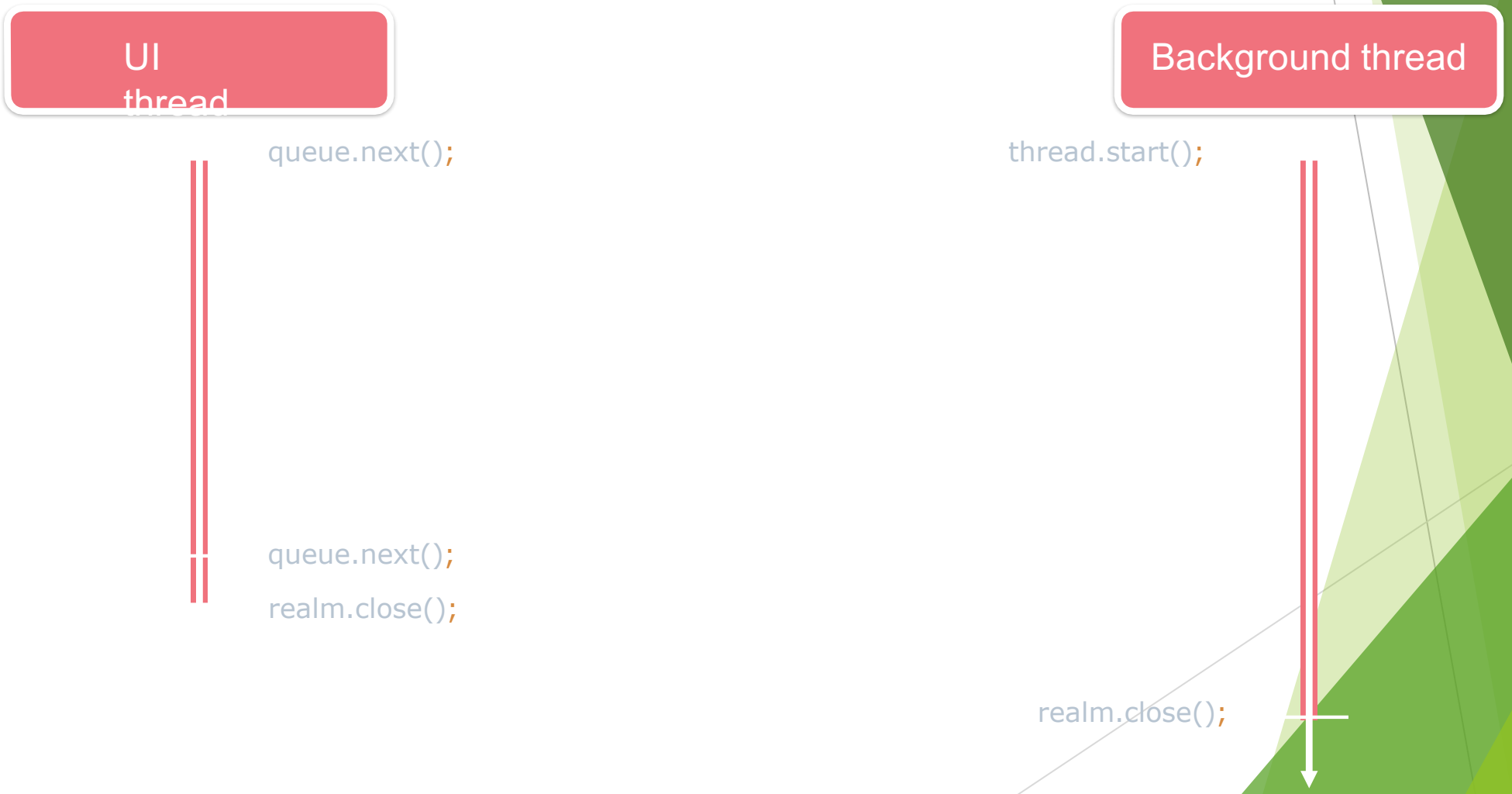
```
queue.next();
Realm realm = Realm.getDefaultInstance();
```

```
thread.start();
```

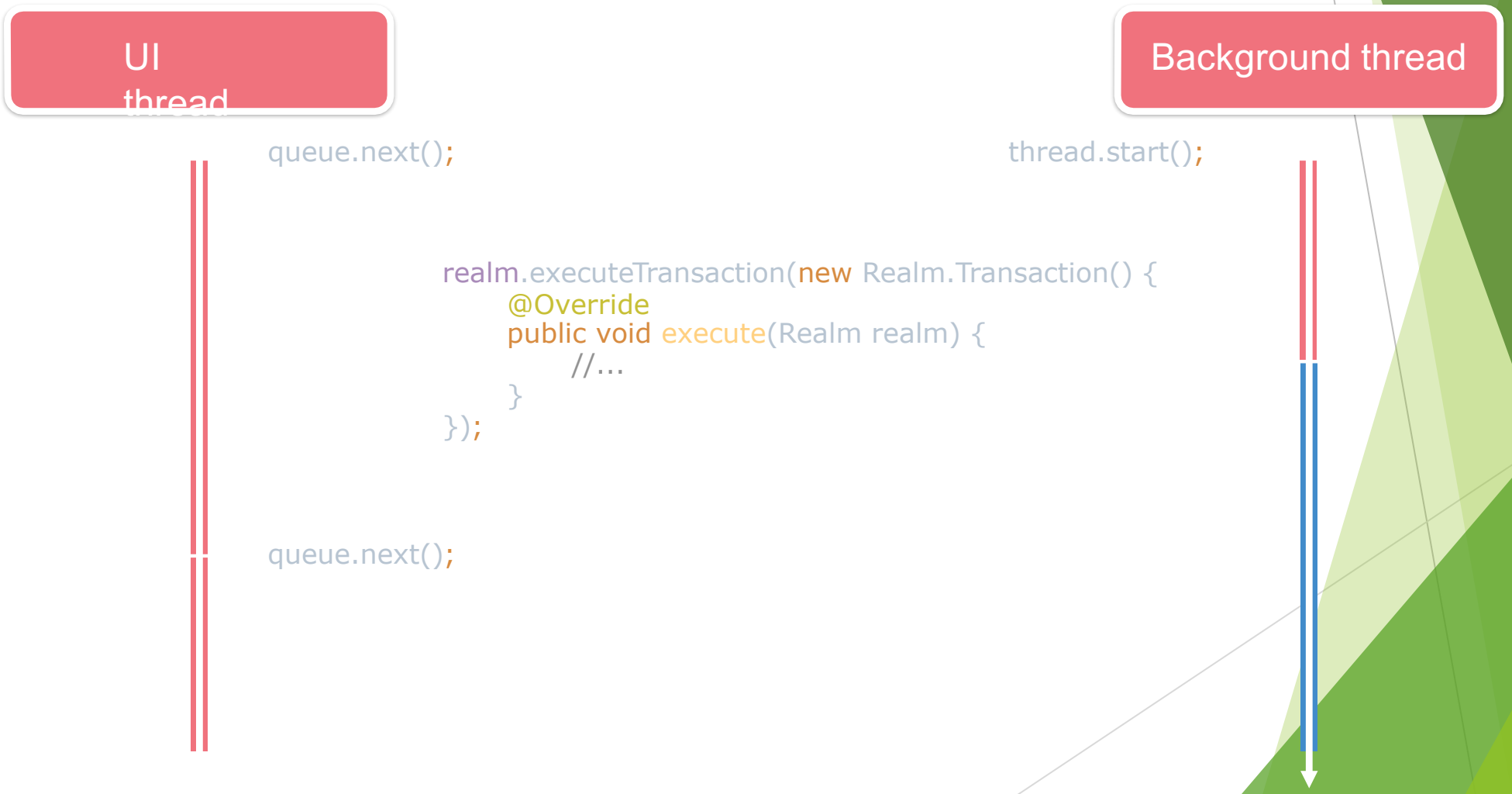
```
Realm realm = Realm.getDefaultInstance();
```

```
queue.next();
```

Consistency in Realm



Consistency in Realm



Consistency in Realm

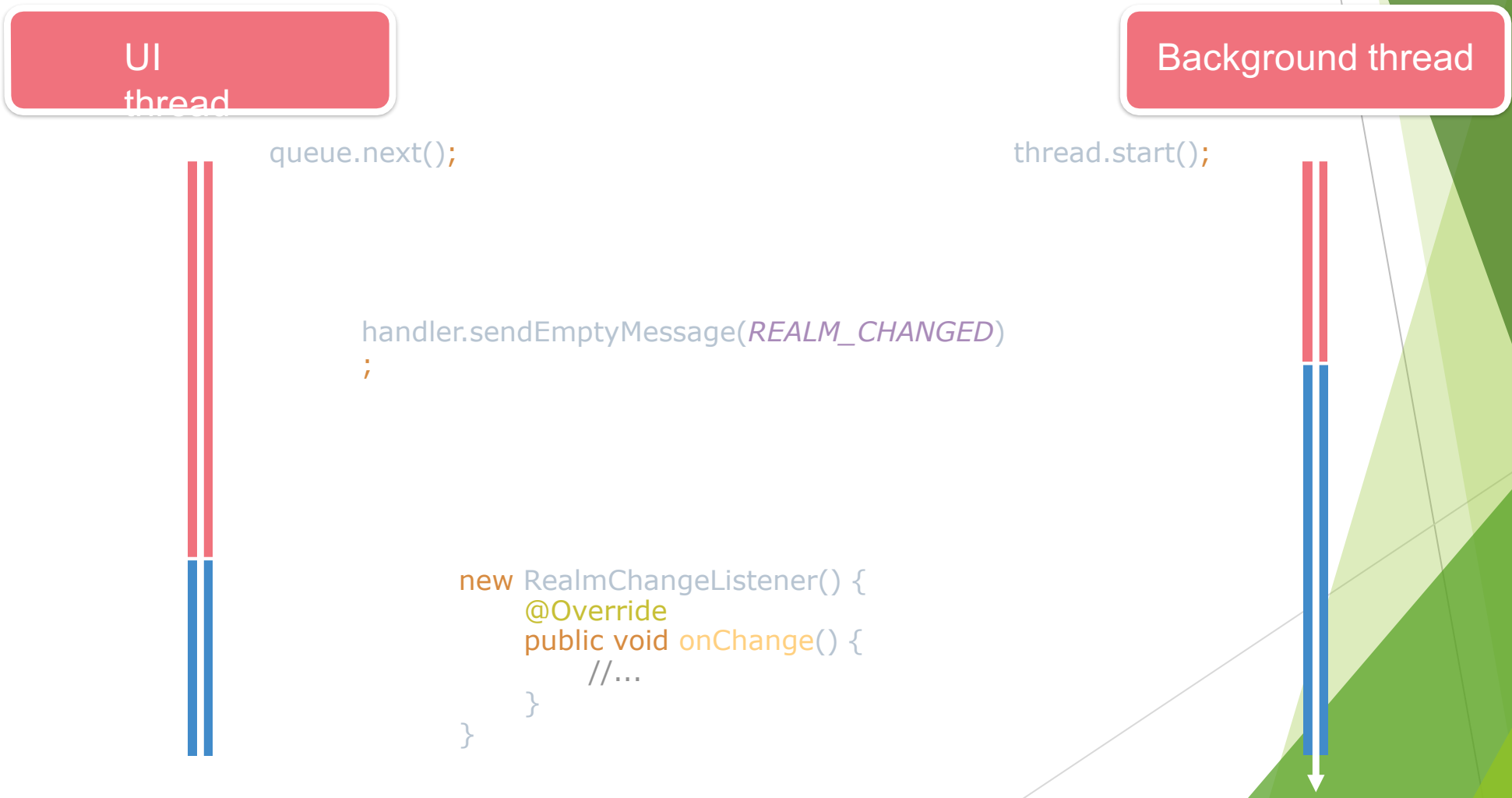
UI thread

Background thread

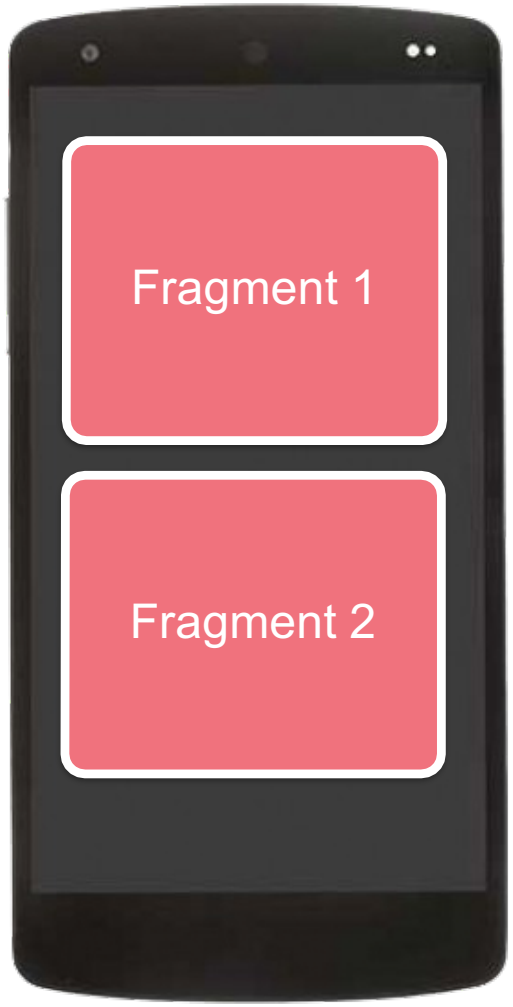
```
queue.next();

@OnClick
public void buttonClicked() {
    Person p = realm.where(Person.class).findFirst();  int
    ager = p.getAge();


    String name = p.getName();
}


queue.next();
```

```
thread.start();
```

Consistency in Realm



Consistency with Realm




```
new RealmChangeListener() {
        @Override
        public void onChange() {
                invalidateViews();
        }
}
```

Realm today

- Moving towards 1.0
- Used in well more than 500M installations.
- Still need to solve hard problems:

  Interprocess cursors, move objects across threads, Parcelable.